1985

Limit analysis of transversely loaded grillages using linear programming.

Harvey, Gerald A.

DEPARTMENT OF OCEAN ENGINEERING
MASSACHUSETTS INSTITUTE OF TECHNOLOGY
CAMBRIDGE, MASSACHUSETTS 02139

LIMIT ANALYSIS OF TRANSVERSELY LOADED GRILLAGES

BY

GERALD A. HARVEY

COURSE 13A
JUNE 1985

S.M. NAVAL ARCHITECTURE & MARINE ENGINEERING
S.M. MECHANICAL ENGINEERING

LIMIT ANALYSIS OF TRANSVERSELY LOADED GRILLAGES
USING LINEAR PROGRAMMING

BY

GERALD A. HARVEY
B.S. Elect. Eng., U.S. Naval Academy
(1972)

SUBMITTED TO THE DEPARTMENT OF
OCEAN ENGINEERING
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREES OF

MASTER OF SCIENCE OF
NAVAL ARCHITECTURE AND MARINE ENGINEERING
and
MASTER OF SCIENCE IN MECHANICAL ENGINEERING
at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY
June, 1985

LIMIT ANALYSIS OF TRANSVERSELY LOADED GRILLAGES USING LINEAR PROGRAMMING

by

GERALD ALAN HARVEY

Submitted to the Department of Ocean Engineering on December 18, 1984 in partial fulfillment of the requirements for the Degree of Master of Science in Naval Architecture and Marine Engineering and Master of Science in Mechanical Engineering

ABSTRACT

A computer program that can determine the collapse load of a transversely loaded grillage that allows for various types of loading and failure modes is presented. The program allows the user to constrain particular grillage members to have fully developed plastic hinges at the location of maximum moment which gives the program the capability to study redundancy of transversely loaded grillages. The lower bound theorem of limit analysis and linear programming techniques are used to determine a grid load factor and provide a failure mode and associated bending moments. A user's guide and program explanation are provided. An example grillage is studied using this program and the load factor and redundancy is discussed.

Thesis Supervisor: Dr. Paul Xirouchakis

Title: Associate Professor of Ocean Engineering

ACKNOWLEDGEMENTS

I would like to thank Professor Paul C. Xirouchakis for his invaluable advice and constructive criticism of my work. I am indebted to the American Bureau of Shipping and the M.I.T. Sea Grant Office for their support of this project. I would also like to express my appreciation to my wife, Patricia, for her enthusiastic support during my studies at M.I.T.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

INTRODUCTION

The capabilility to easily and effectively study a structure from the stand point of redundancy would significantly contribute to the tools available to engineers for calculating safety factors and provide regulatory bodies an inexpensive way to verify or develop safety factors with analytical basis.

Strengthening of ship structures has been an area of a significant amount ot work. Ice loading is particularly threatening to ship safety and also has the characteristic of very large localized or point loads. The program presented here is ideal for the study of local loading or failure.

Previous work done on grillages that used similar techniques to determine the load carrying capability and possible failure modes was done by Tait and Hodge (1) in their development of a computer program called BEAMPLT that utilizes linear programing techniques to find a solution to transversely loaded grillages. This work was later expanded upon to provide a simple method of computing failure modes and load factors under various patch loads by Abbott (2).

Hodge (1), in his paper on transversely loaded grillages, demonstrated the feasibility of using linear programing to calculate a failure mode and load carrying

capacity (load factor) for transversely loaded grillages. His work allowed for several types of boundary conditions. These conditions included simply supported, clamped, and cantilevered. The loading scheme was limited to point loads on each node.

Abbott (2) used this approach to solve for various types of patch loads on a grillage. He wrote a computer program that employed the same techniques as Hodge used. He employed linear programing techniques for calculating a failure mode and load factor for a given grillage. The patch loads that he was interested in were those that could be associated with relatively small area ice loading on the side of a ship. These patch loads were transferred to the nodes of the grillage using a simple lever principle.

Both Hodge (1) and Abbott (2) allowed failures to occur only at nodes in the grillage. Hodge's scheme allowed loading only at the nodes. Abbott's method was adequate to estimate failure with reasonable accuracy, but his assumptions in transferring patch loads into nodal loads was simplistic and did not allow for point loads off of a node. Abbott's approach followed Hodges very closely and much programing effort was made to allow for all the boundary conditions that Hodge had programmed for.

The computer program presented here is simplified for implementation on a VAX computer with one of the standard International Mathematical and Statistical Library's linear

programing routines. Linear programming is used because it is fast, inexpensive and easily adaptable to many geometries. I have allow for only those boundary conditions that are frequently encountered in ship structures, that is clamped or simple supported. The capability to handle point or linear line loads on any grid member has been introduced. The ability to handle a uniform load over the entire grillage is also important and should be developed. The most important aspect ,however, is the capability of adding additional constraints to the linear programing routines that will allow for failures at other than a node. An iterative approach to obtain a final solution to this general loading scheme is employed. One of the many types of problems that this approach is well suited for is redundancy study. In developing the capability to introduce local failures and the ability to handle large local loads the program finds particular suitable use in analyzing ice loaded ships structures or slow collisions. I will demonstrate its applicability for this by discussing the redundancy of a grillage of a ship's side that is loaded as one might expect to encounter in ice loaded ship structures.

CHAPTER 1

MODELING CONSIDERATIONS

The problem of transversely loaded grids covered with plating is a complex one that, except for very simple cases does not have a closed solution. There are several aspects of this problem that can be easily simplified without significantly effecting the end results. In ship structures it is the goal to obtain a reasonable bound for the ultimate strength of a given structure in order to minimize its weight and in doing so maximize its efficiency. It is well known that the thin skin plating adds little to the transverse strength of a grillage and can easily be included as an effective width calculation in determining the fully plastic bending moment of the individual members of the grillage. As such, this paper will deal entirely with the strength of the grid. When a grid is deflected, twisting moments are introduced in grid members. For small deformations, these twisting deformations are relatively small and will be ignored. It is also assumed that shear force does not contribute to the breakdown at any plastic hinge. For analysis, all grid members are assumed to be perfectly elastic and undergoing only small deformations. The program will identify a lower bound for ultimate collapse and give a possible failure mode.

The problem reduces to a limit analysis of grillages loaded with a variety of transverse loads. In order to

apply linear programming techniques each load must be reduced to corresponding point loads at nodes. This reduction does not adversely affect the outcome since it is at these intersections that the interactions occur. The behavior of the member itself is determined later using the moments calculated at each node as the end moments for the given member. Once the load reduction is complete the lower bound plastic load limit is found using a standard linear programming routine.

As a simple example consider a simple supported nine node grid consisting of six members arranged sysmmetrically as shown in figure (1). Applying a concentrated load F at node number 5, the center node, will produce a failure mode as shown. The circles indicate fully developed plastic hinges. Twisting does not occur in this example because of symmetry. This is not the general case. The effect of torque in normally encountered structures is relatively small when compared to bending, and in ignoring twisting moments results will be conservative. Tait (1) has shown that for typical I sections the error is less than 0.1 percent for the types of grids generally considered. For long narrow less symmetrical grids the error may be as large as 8 percent.

Determining the work done in creating the deformation pattern shown in figure 2., which is the only symmetrical failure mode possible for this geometry, the lower plastic

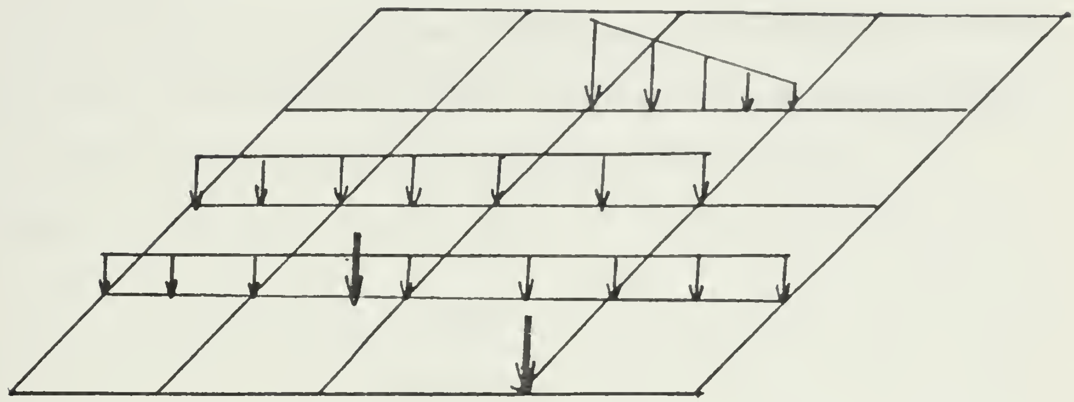


Figure 1. Typical grillage with general loading

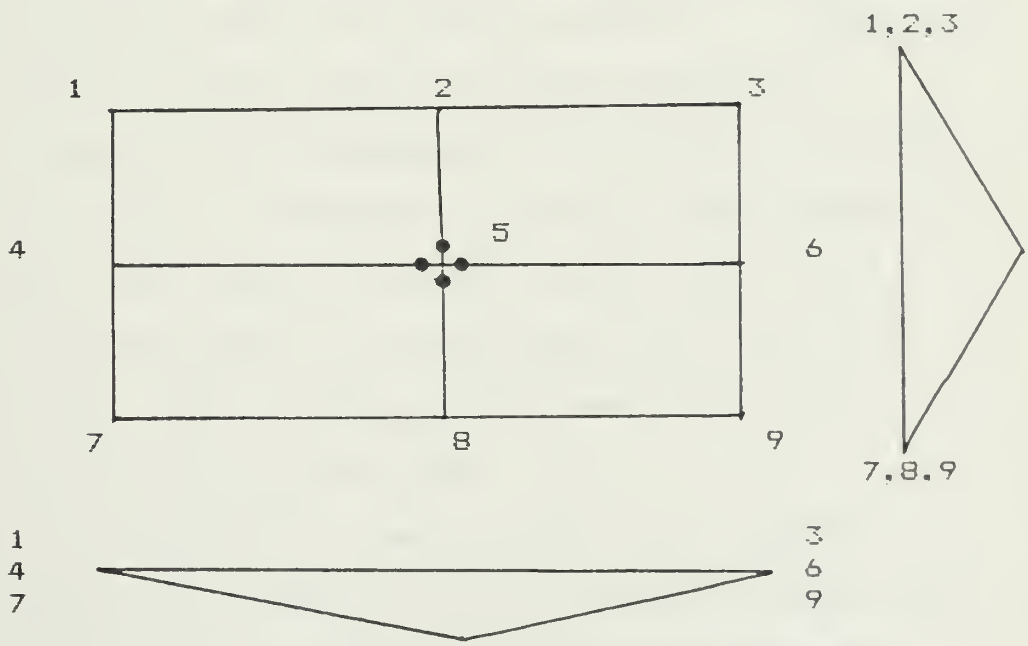


Figure 2. A nine node grid collapse mode and deformation profiles

limit for collapse can be determined. If the angle of rotation is θ then the transverse load is displaced by θL where L is the length of each member. And, the external work is $F\theta L$. Each hinge shown undergoes a rotation of θ so the total work performed is $4Mo\theta$. By setting the two values equal to each other the total collapse load can be calculated.

$$FL\theta = 4Mo\theta$$

$$F = 4Mo/L$$

The program GRIDS presented here produces identical solutions. The results of other more complex examples completed by Hodge (3) and Tait (1) have also been duplicated and the results have been identical. GRIDS is a FORTRAN computer program that uses plastic methods of analysis to determine the response of a grillage subjected to a general scheme of transverse loading. The program can handle linear line loads on a member, point loads at any location on the grillage or a distributed load over the entire grillage. This program can be used to find the plastic load limit of any grillage.

GRIDS is a relatively simple program that employs a linear programing technique to solve a system of equations that are constrained by the limit of a fully developed plastic moment in a member. A program developed by Tait and

Hodge (1) produced a similar solution but was limited in the type of loading that it was able to handle. It allowed only point loads at beam intersections. Abbott (2) modified Tait's and Hodge's program and introduced a simple geometric method of reducing patch loads to nodal loads (loads at intersections) and then applied Tait's and Hodge's method directly to arrive at a solution. GRIDS uses the same technique combined with the specific loading scheme to check individual grillage members for potential local failures and then employ an iterative process in narrowing the bound and failure mode based on the points in the grillage not on the nodes that are most likely to fail.

Program Input and Problem Approach

The input data used by GRIDS consists of the specific geometry, number of beam intersections, beam yield moments, lengths of the beams, boundary conditions, and specific data for all the loads. The program is limited to one point load per node, one additional point load per member and one linear line load per member.

Once all the loading data is entered, equations of equilibrium are developed for each member. In reducing these equilibrium equations for each member the assumption is made that no local failure of the member has occurs. This is verified with the output from the first iteration of GRIDS. If a particular member is found that has exceeded its yield moment then the moment at the location of this

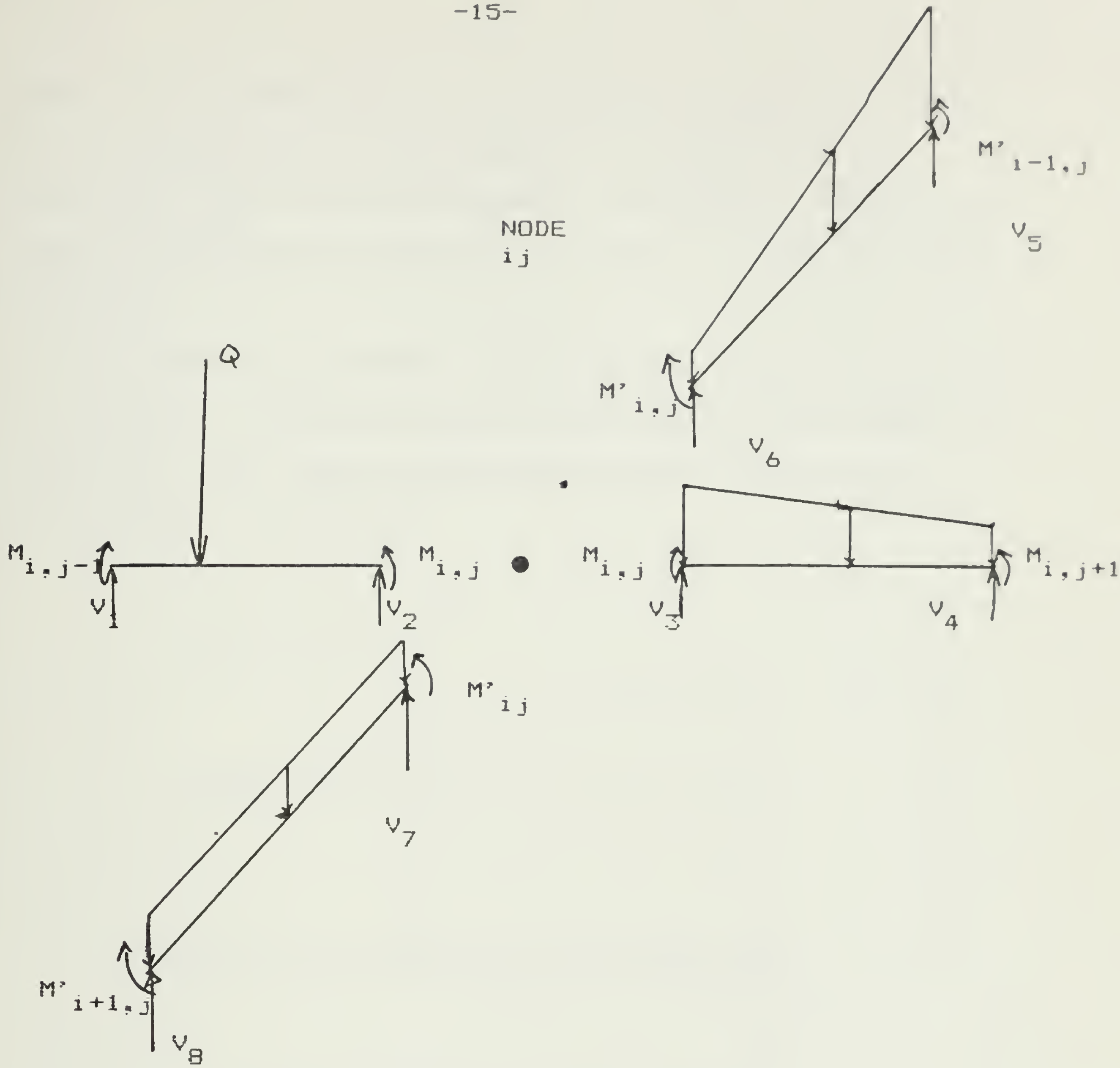


Figure 3. Free Body Diagram for a Node and Surrounding members (general loading condition)

occurrence is constrained. An additional equation is added to the system for each such member. A new solution is determined which includes these new constraints. Figure (3) shows an equilibrium diagram for a node in a general loading condition.

In order to calculate a solution the following approach is followed. The shear forces at the ends of each member is solved for and a moment balance performed. For example, taking member (i) we have:

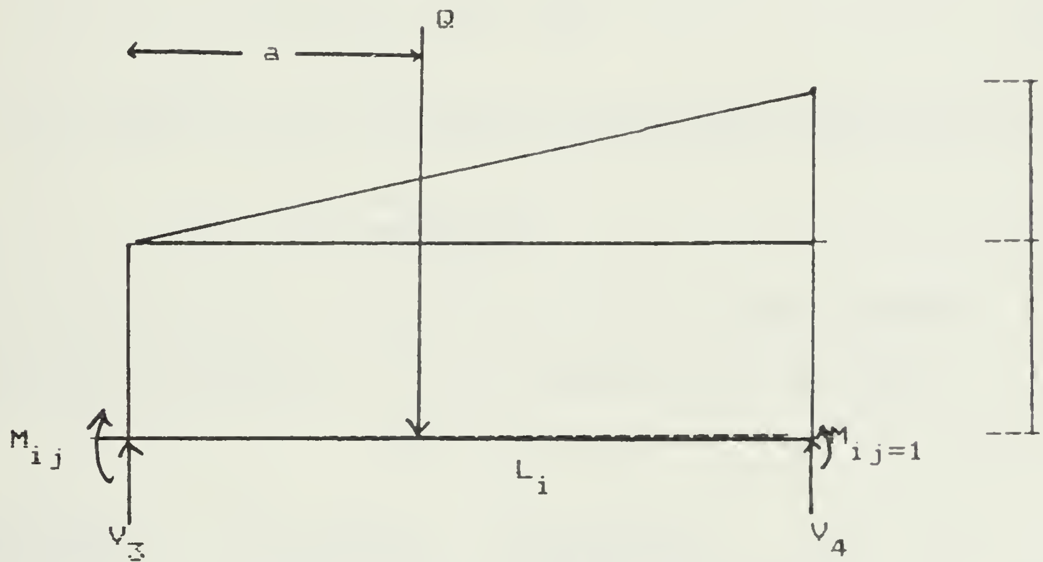


Figure 4. Equilibrium of a Member

$$V3 = \frac{M_{ij+1} - M_{ij}}{L} + \left[1 - \frac{a_i}{L}\right] Q + \frac{W_i L_i}{2} + \frac{P_i L_i}{3} \qquad (1)$$

$$V4 = \frac{M_{ij} - M_{ij+1}}{L} + \left[\frac{a_i}{L}\right] Q + \frac{W_i L_i}{2} + \frac{P_i L_i}{6} \qquad (2)$$

The total shear at a node will be the sum of the shears of the members that meet at that node. The only unknowns in these equations are the end moments. Combining the load terms and end shears into a single load for each node f_{ij} we solve for the vertical and horizontal moments at each node.

Referring to figure (3) we have from equilibrium of force at node ij ;

$$f_{ij} = V2 + V3 + V6 + V7 \qquad (3)$$

and from equilibrium of moments across each member we have

$$V3 = V4 = \frac{M_{ij+1} - M_{i}}{L} \qquad (4)$$

Similar expressions are obtained for all other members. Substituting equation (4) into equation (3) we have

$$\frac{M_{ij} - M_{ij+1}}{L_{j-1}} + \frac{M_{ij} - M_{ij+1}}{L_{j}} + \frac{M_{ij} - M_{i-1j}}{L_{i-1}} + \frac{M_{ij} - M_{i+1j}}{L_{i}} = PP_{ij} \; f_{ij} \qquad (5)$$

which satisfies equilibrium of node ij. Similar equations are developed for each node.

The program GRIDS constrains each moment to satisfy

$$-Mo < M_{ij} < Mo \quad (6)$$

$$-Mo' < M'_{ij} < Mo' \quad (7)$$

where Mo is the yield moment of a particular member. The yield moments of symmetrical beams can easily be calculated from the geometry and material properties of the beam.

Now GRIDS applies a linear programming routine to the system to find a solution. To do this the program will maximize the load factor PP, where PP is the multiplier of each of the applied loads required to induce collapse. In maximizing PP a maximum number of moments M_{ij} and M'_{ij} will reach the imposed constraint $\pm$ Mo.

The linear programming problem seeks to minimize an objective function that is a linear function of unknowns, subject to constraints. These constraints consist of linear equalities and inequalities. In the standard form the inequalities can be expressed as equalities and the problem becomes:

$$\text{minimize} \quad f(X1, X2, \ldots, Xn) = C_1X_1 + C_2X_2 + \ldots C_nX_n \quad (8)$$

$$\text{subject to} \quad \begin{aligned} a_{11}X_1 + a_{12}X_1 + \ldots + a_{1n}X_n &= b_1 \\ a_{21}X_1 + a_{22}X_1 + \ldots + a_{2n}X_n &= b_2 \\ &\vdots \\ &\vdots \\ a_{m1}X_1 + a_{m2}X_1 + \ldots + a_{mn}X_n &= b_m \end{aligned} \quad (9)$$

The coefficients of the constraint equations and of the

objective function are known values and the X_i's are unknowns.

Since our problem is to maximize PP it should be noted that minimizing f(X1,X2,X2,....,Xn) is equivalent to maximizing -f(X1,X2,X3,....,Xn). A detailed description of linear programing can be found in reference (4).

The objective function of our problem becomes:

$$C_1X_1 + C_2X_2 + \ldots\ldots C_nX_n = PP \tag{10}$$

where the X's are given by

$$X_n = M_{ij} + M_o \tag{11}$$

so that the inequalities (6) and (7) become

$$0 < X_n < 2M_o \tag{12}$$

$$0 < X_n < 2M'_o \tag{13}$$

CHAPTER 2

PROGRAM OPERATION

GRIDS is a Fortran program written in a modular fashion that makes maximum use of separate subroutines and common memory. This approach makes the program much easier to understand and gives it the flexibility to be easily changed.

The program has some specific limitations. All horizontal members and their maximum yield moments must be the same. All vertical members and their maximum yield moments must be the same. However, vertical and horizontal members need not be the same. Boundary conditions may only be clamped or simple supported. But, the boundary condition of each side is independent of the others. Point loads may be applied at any point of on the grillage, but there must be only one point load per member with one additional point load allowed at each node. One linear line load per member is allowed. All loads must be applied perpendicular and transverse to the grillage.

The main program GRIDS assimilates major variables and input and sequences the subroutine operation. There is only one set of significant calculations performed in the main program. The calculation consists of determining the dimensions of the arrays that the IMSL linear programming routine ZX3LP uses and could not be done in the subroutine that actually calls ZX3LP.

Each of the program's functions is controlled by an independent subroutine. The controlling program GRIDS takes the user input for the grillage to be studied and assigns variables for the boundary conditions and geometry of the structure. Subroutine INTPL initializes the load array. MINT uses the specific boundary conditions and initializes the output load arrays HMNT(i) and VMNT(i). The index i is for the ith node. These arrays are filled with ones and zeros. All nodes that are known to have a zero moment because of the boundary conditions are assigned a zero, all other unknown moments are assigned a one. These arrays are used to determine the total number of unknowns and are used as multipliers in Subroutine CONST to ensure that the boundary conditions are handled properly (ie. zero moments at simple supports). Later they are used to store the nodal moments calculated in the linear programming routine.

Subroutine ENTRL handles assigning all loads to the proper arrays and decomposes them into nodal loads for use. Each load must be stored so that individual beam moment distributions can later be calculated by Subroutine BEAMCK.

Once the loads are entered the user decides if yield moments are allowed to form in the members away from the nodes. If he chooses to allow additional plastic hinges then the subroutine CONST is called and the additional constraint equations are written. These equations may be generated automatically in which case they are based on the

grillage scheme and an initial check of each member. They can also be generated one at a time manually. In this case the user inputs which member is to have the additional constraint and the location on the member that is expected to develop the plastic hinge. In addition, if the manual mode is chosen, the constraint equation can be written normally as a likely failure point or it can be written as if the point has already failed due to some very large load (in this case it becomes an equality constraint). These possibilities are the heart of the program and are discussed in detail in the body of this paper.

Subroutine EQNS is used to calculate the nodal equilibrium equations that are sent to IMSL's ZX3LP. Subroutine Fixer transforms all of the equality and inequality constraints into arrays that ZX3LP can use and calls ZX3LP. It then translates the results into horizontal and vertical moments at each node.

Once this is complete the moments at each node and the specific loading scheme are used by BEAMCK to determine the maximum moment and its location on every loaded member. If automatic constraints was chosen at the beginning of execution then the program will automatically constrain any moment found to be greater than the yield moment and iterate until a solution is found where no moment exceeds the yield moment or until the maximum number of iterations is reached.

Subroutine OUTPUT writes all grid information to a data

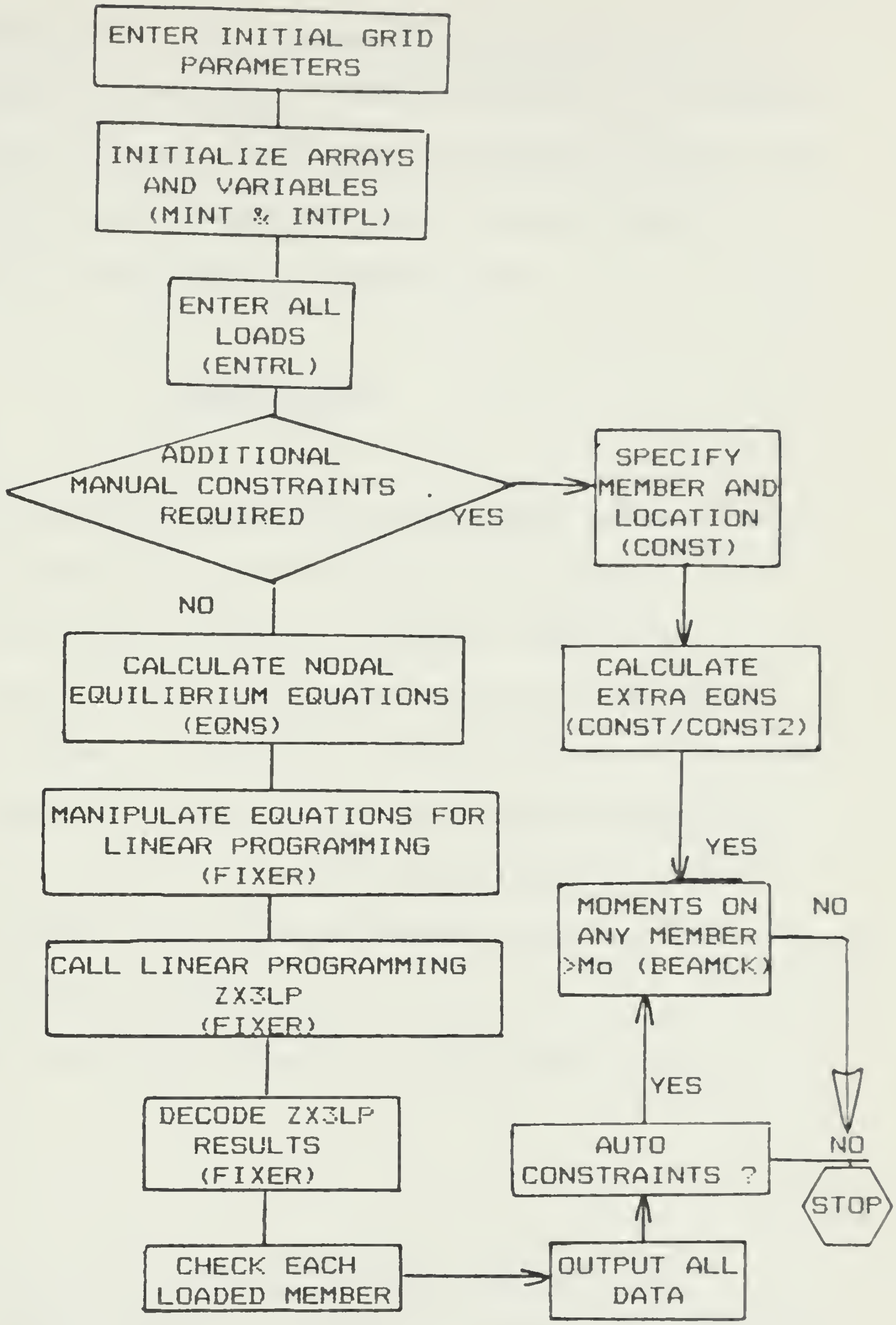


Figure 5 Macroscopic Program Flow Diagram

file named MOMENTS.DAT. All inputs are summarized and a ZX3LP error number is provided. Output includes all members ind moments, nodal loads (from decomposed general loads) and the maximum moment and location for each loaded member. Figure 5 shows the macroscopic program flow.

SUBROUTINES

MINT AND INTPL

MINT initializes the nodal bending moment arrays. Any moment that is known to be zero because of boundary conditions are set to zero and all others, which are unknowns, are set to one. The master program GRIDS then calculates the number of unknowns in the problem by counting the non-zero elements in the array. These ones and zeros are used as multipliers to cause the coefficients in the equilibrium equations calculated in CONST to go to zero to comply with boundary conditions.

INTPL sets the starting value of each nodal load to zero.

ENTERL

ENTERL is the subroutine that makes load variable assignments. Its operation is straight forward and requires little explanation. Some error traps are built in to ensure all loads are in fact valid from the standpoint of location. There is no trap however to ensure that the restrictions on the numbers of various loads are observed.

If these restrictions are not observed the calculated

PROGRAM VARIABLES

Arrays

PTLOAD(i)	The total load for node (i)
II(i)	The Y coordinate of node (i)
JJ(i)	The X coordinate of node (i)
LD1ND(i)	The reference node number of point load (i)
D1VAL(i)	The magnitude of point (i), a negative value indicates the point load is on vertical member, a positive value indicates the load is on the horizontal member
D1LOC(i)	The distance of point load (i) from the reference node
LD2ND(i)	The reference node number of line load (i)
D2AVL(i)	The magnitude of line load (i) at the reference node, a positive value indicates the load is on the horizontal member
D2BVL(i)	The magnitude of line load (i) at the node opposite to the reference node, a positive value indicates the load is on the horizontal member
LD3ND(i)	The reference node number of distributed load (i)
D3VAL(i)	The magnitude of the distributed load
VMNT(i)	The value of the vertical moment at node (i)
HMNT(i)	The value of the horizontal moment at node (i)
BHKMNT(i)	Maximum moment on horizontal member of reference node (i)
BVKMNT(i)	Maximum moment on vertical member of reference node (i)

Table 1 Program Variables

BVKLOC(i)	Location of max moment on vertical member of reference node (i)
BHKLOC(i)	Location of max moment on horizontal member of reference node (i)
XCNST(i)	Location of manual constraint for reference node (i)
NCNST(i)	Reference node of manual constraint (i)
FOG(i,j) \ A2(i,j) ______ B1(i) / B2(i) /	These are dummy arrays for manipulating data for use by the linear programming subroutine

<u>Variables</u>

NODESH	Number of beam intersections in the horizontal direction (nodes)
NODESV	Number of beam intersections in the vertical direction (nodes)
NODTOT	Total number of beam intersections, all edges have beams along them
VLEN	Length of the vertical members, distance between nodes
HLEN	Length of the horizontal members, distance between nodes
LPT	Number of point loads added to the grillage
LNLDN	Number of linear line loads added to the grillage
LDIS	Number of distributed loads added to the grillage
ITOP	Top edge boundary condition
ILFT	Left edge boundary condition

Table 1 Program Arrays and Variables (continued)

IRGT	Right edge boundary condition
IBTM	Bottom edge boundary condition
PP	Grid load factor
IER	Linear programming routine error number
IEQNM	Number of non-zero moments in the grid
NODEEQ	Number of nodal equations required to obtain a solution
HORZMO	The fully plastic moment of the horizontal grid members
VERTMO	The fully plastic moment of the vertical grid members

Other variables used throughout the subroutines are dummy variables used in calculations or for keeping track of vector sizes and locations.

Table 1 Program Arrays and Variables
(continued)

maximum moment and location on each member will not be correct.

There are several arrays for each load type. The total number of each type of load is kept track of with load counters. LPT is the total number of point loads entered and LNLDN is the number of line loads. LD1ND and LD2ND store the reference node numbers of each point load and line load respectively. D1VAL and D1LOC store the magnitude and location (distance from the reference node) of each point load. D2AVL and D2BVL contain the end point magnitudes of each line load.

Table (2) gives a summary of load array functions.

All array elements are indexed by load number for each specific type of load. Each load type has an array that contains the reference nodes associated with the load numbers. The sign of the load magnitude determines which member, either horizontal for positive or vertical for negative, the load is associated with. The reference node is always the node to the left or above the loaded member. Figure 6 shows the program flow for entering loads.

EQNS

Subroutine EQNS writes the nodal equations of the grillage and writes them to an array in a format that subroutine FIXER uses later to assimilate all of the required data into the proper format. EQNS calculates the

Point Loads	LD1ND(i)	Contains the reference node number for point load number i
	D1VAL(i)*	Contains the magnitude of load number i
	D1LOC(i)	Contains the location of (distance from the reference node) load number i
Line Loads	LD2ND(i)	Contains the reference node number for point load number i
	D2AVL(i)*	Contains the magnitude of load number i at the reference node
	D2BVL(i)*	Contains the magnitude of load number i at the opposite node
odal Loads	PTLOAD(i)	Store the total nodal load (decomposed general loads) that the equilibrium equations are written from

* These values are set to the negative to indicate the load is on the vertical member associated with the reference node and are left positive to indicate the load is on th horizontal member.

Table 2. Load Array Function

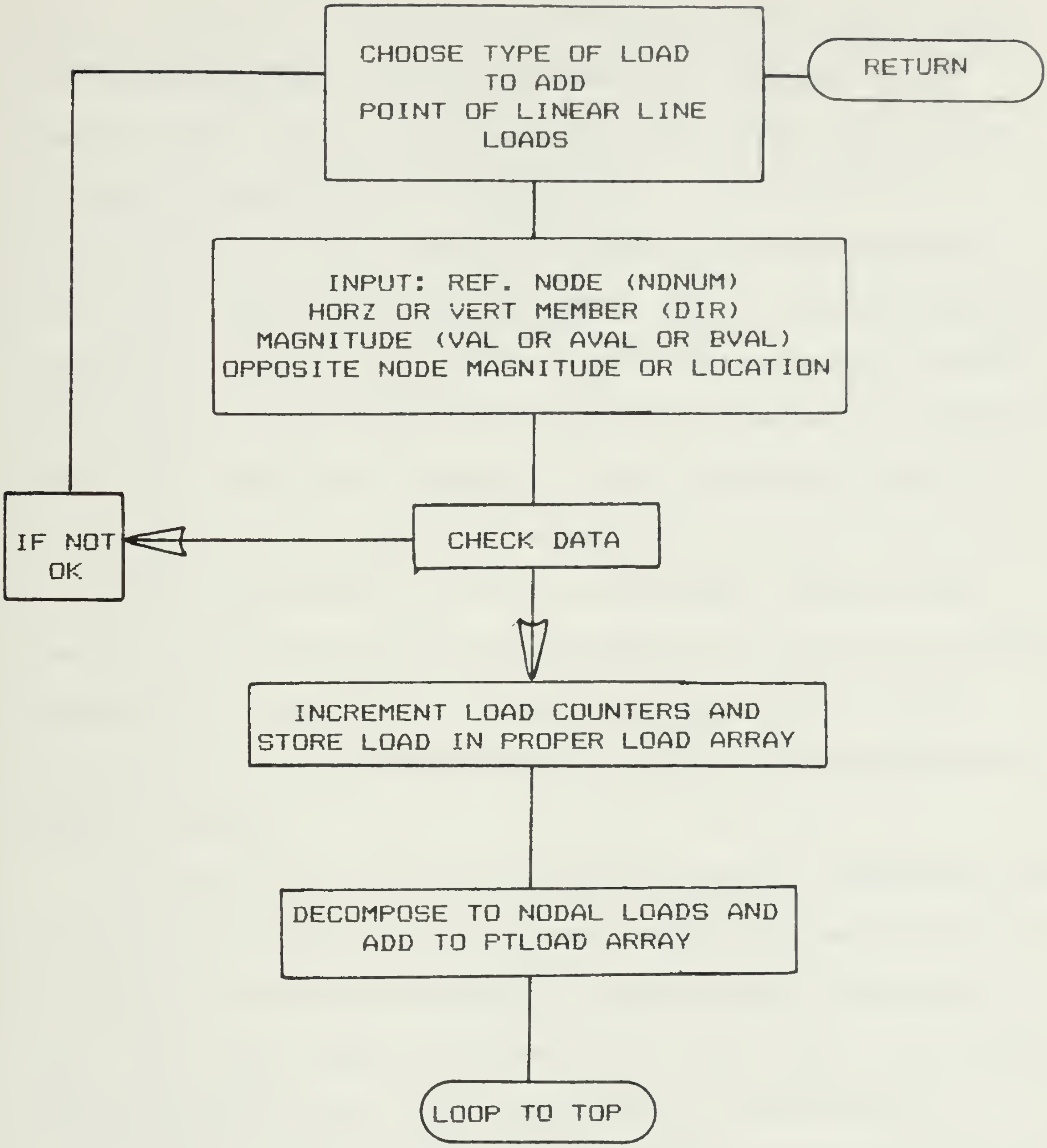


Figure 6 Program Flow for Entering Loads (ENTERL)

coefficients of the equilibrium equations for each member. Since the unknowns are the end moments, a single linear equation that contains each unknown throughout the entire grillage is generated for each member. The result is a vector with twice the number of nodes plus one elements. Since each member has only two ends, all of the coefficients except two turn out to be zero. Only the two coefficients associated with the nodes that bound the member will be non-zero. The additional unknown is the load factor which is not constrained but will be maximized. The left half of this array represents all of the horizontal moments and members while the right half represents the vertical moments and members. Figure (7) shows the construction of the vector matrix for all equations written by both subroutines EQNS and CONST.

In Figure (7) the first example equation represents the equilibrium equation to the member located between the node in the third and forth columns or node numbers three and four. The last column represents the load factor and the B vector represents the right hand side of the equation. Since the non-zero elements are located in the left half of the matrix we know that this equation represents a horizontal member. The second equation has non-zero elements in the right half of the matrix so it must be for a

```
              <------------- A2 Matrix ---------------->     < B >
 /!\     |<--total # nodes-->:<--total # nodes-->:  |     |     |
  #      | 1 2 3..........n  :  1 2 3..........n :  |     |  M  |
         |                   :                   :  |     |  a  |
  u      |Horizontal Section : Vertical Section  :  |     |  t  |
  n      |                   :                   :  |     |  r  |
  k      | Example equations :                   :  |     |  i  |
  n      |                   :                   :  |     |  x  |
  o      | 0 0 1 3 0 0 0 0 0 : 0 0 0 0 0 0 0 0 0 :  |     |     |
  w      | 0 0 0 0 0 0 0 0 0 : 0 0 0 3 0 0 0 3 0 :  |     |     |
  n      |                   :                   :  |     |     |
  s      |                   :                   :  |     |     |
  !      |                   :                   :  |     |     |
 \!/     |                   :                   :  |     |     |
```

Figure 7.

vertical member. Since all of the nodes of the grillage are numbered from left to right we know that all of the horizontal members will be represented by coefficients in adjacent elements. Likewise, vertical members will be represented by elements in the right side of the matrix that are separated by the number of horizontal node in the grillage. Figure 8 shows the flow diagram for EQNS.

<u>CONST and CONST2</u>

Subroutine CONST constructs additional constraint equations in order to refine and clarify the problem solution. Specified members are selected for these additional constraints. Coefficients of the equilibrium nodal equations for each of these specified members in the grillage that has unknown end moments are then calculated.

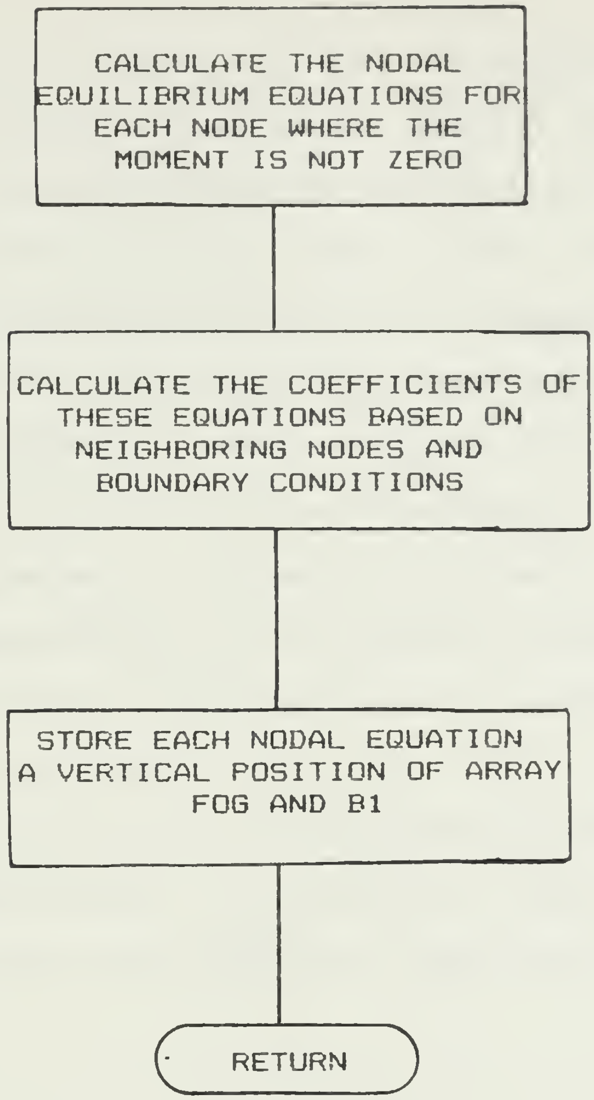


Figure 8 EQNS Subroutine Flow Diagram

It has Automatic or manual options. If automatic is selected the program calculates the maximum moment found in a member as the sum of the linear combination of the member's end moments as calculated from the initial nodal loading and the particular loading of the member. The coefficients of the nodal equations involving that member are then calculated using the location of the maximum moment found during this initial look at the member. In manual, the location of the moment is chosen by the user. In manual the user also chooses whether or not the extra constraint is to have a moment already developed (ie prior damage) or it is to be handled as a limit.

CONST2 operates exactly as subroutine EQNS to write an Array of Coefficients to be used by subroutine FIXER. Figure 9 shows the flow diagram for CONST and CONST2.

FIXER

Subroutine FIXER gathers the known data and converts it into a form that can be used by the IMSL linear programming routine, ZX3LP. It writes an identity matrix that acts as the list of inequality constraints and it uses the arrays produced in EQNS and CONST2 to write the equality constraints and additional constraints. All of these are combined in a single matrix that is used by ZX3LP. See Appendix B for details of this matrix. ZX3LP is then called. FIXER then converts the solution returned from

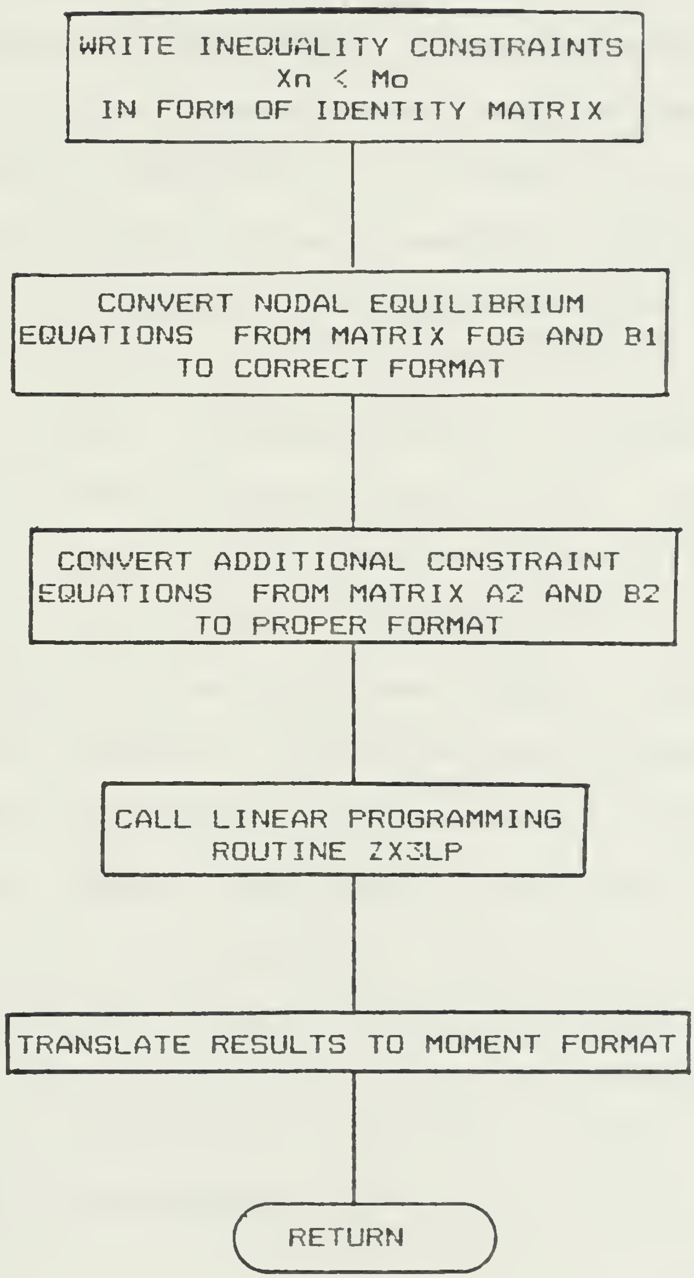


Figure 10 Subroutine FIXER Flow Diagram

BEAMCK

Subroutine BEAMCK calculates the maximum moment and location for every loaded member. It accomplishes this by looking at each node and gathering the loads that are referenced to it. If no loads are referenced there will be no calculations for that node. It then separates the loads into vertical and horizontal members and assigns them to dummy variables. Equations for the moment as a function of position are available for each possible loading condition and position on a member. There are four possibilities. The linear line load may be either increasing or decreasing going away from the reference node and the moment must be calculated for either side of a point load. A constant line load will fir either type of linear line load equation. These equations are derived in Appendix C.

The program applies the proper equation and calculates the moment and location. The results are stored in four arrays. BHKMNT and BHKLOC store the moment and location for the horizontal member referenced to a node and BVKMNT and BVKLOC store this information for the vertical members. It should be noted that subroutine CONST uses identical logic to determine which additional constraint equations to write when automatic constraints is selected. Figure 11 shows the subroutine flow for BEAMCK.

OUTPUT

Subroutine OUTPUT consists almost entirely of write and

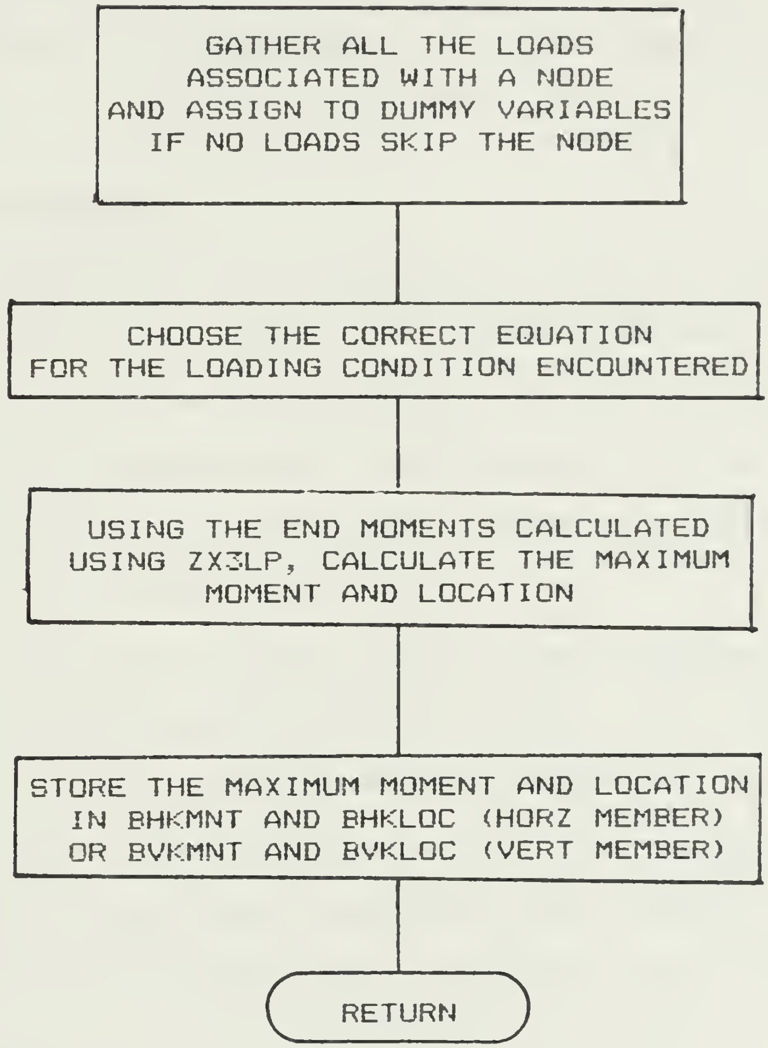


Figure 11 BEAMCK Subroutine Flow Diagram

format statements that provide a hard copy of the grid characteristics, loading and the solution. A ZX3LP error number is also provided which can be referenced in Appendix B if problems are encountered. A sample output is shown in Figure 12.

RUNNING THE PROGRAM

Once GRIDS and its subroutines, including the IMSL routine ZX3LP and ZX3OP (ZX3OP is used by ZX3LP), are installed on the operating system, the program can be run. It may be run interactively using keyboard input or it may be run using an input data file. Appendix B shows how to create an input data file.

The following listing is the actual program dialogue as seen on the terminal when running Grids. User responses have been put in square brackets for clarification. The listing below is typical but not all inclusive. If automatic constraints is selected, the inputs that concern manual constraints is not asked for. All other inputs are similar.

```
[$ RUN GRIDS]
PROGRAM RESTRICTED TO A MAXIMUM GRID OF 6 X 5
IE 30 NODES TOTAL OR LESS
THIS RESTRICTION IS IMPOSED BECAUSE OF THE MAXIMUM
SIZE OF THE WORK VECTOR USED BY ZX3LP IS CONSTRAINED
BY THE PROGRAMMER.
SEE PROGRAM MANUAL FOR MORE INFO

INPUT NUMBER OF NODES ALONG THE TOP AND SIDE OF THE GRID

[4,4]
INPUT BOUNDARY CONDITIONS FOR THE TOP
```

```
LEFT,RIGHT AND BOTTOM EDGES OF THE GRID
 IN THAT ORDER
INPUT A 1 FOR SIMPLY SUPPORTED EDGE
INPUT A 2 FOR A CLAMPED EDGE
ALL VALUES ARE INTEGERS

[1,1,1,1]
INPUT HORIZONTAL MEMBER LENGTH (REAL NUM.)

[15.]
INPUT VERTICAL MEMBER LENGTH (REAL NUM.)

[15.]
INPUT HORIZONTAL BEAM YIELD MOMENT (REAL)

[30000.]
INPUT VERTICAL BEAM YIELD MOMENT (REAL)

[30000.]
  LOADS ARE ENTERED REFERENCED TO THE NODE
  ABOVE OR TO THE LEFT OF THE LOAD
  ALLOWED LOADING CONFIGURATIONS:
      1. POINT LOADS
      2. LINEAR LINE LOADS ALONG A MEMBER
      3. LOAD ON A PLATE ELEMENT



  ENTER THE TYPE OF LOAD TO ADD
        1 = POINT LOADS
        2 = LINEAR LINE LOAD ALONG A MEMBER
        0 = NO MORE LOADS TO ADD
[1]
THIS ROUTINE ADDS POINT LOADS TO THE GRID.  THERE MAY BE
A MAXIMUM OF ONE POINT LOAD PER NODE PLUS ONE PER MEMBER
THE REFERENCE NODE IS ABOVE OR TO THE LEFT OF THE LOAD
ENTER FOUR VALUES TO DEFINE EACH LOAD
      1ST VALUE = REFERENCE NODE (INTEGER)
      2ND VALUE = VERTICAL OR HORIZONTAL MEMBER
                  1 = HORIZONTAL MEMBER (INTEGER)
                  2 = VERTICAL MEMBER (INTEGER)
                  0 = ON THE NODE (INTEGER)
      3RD VALUE = LOAD MAGNITUDE (REAL)
      4TH VALUE = DISTANCE FROM THE REFERENCE
                  NODE (REAL)(IF THE 2ND VALUE = 0
                  THEN THE 4TH MUST = 0)
ENTER ALL ZEROS WHEN COMPLETED WITH POINT LOADS

[6,1,7.5,7.5]
ENTER FOUR VALUES TO DEFINE EACH LOAD
      1ST VALUE = REFERENCE NODE (INTEGER)
      2ND VALUE = VERTICAL OR HORIZONTAL MEMBER
```

```
                 1 = HORIZONTAL MEMBER (INTEGER)
                 2 = VERTICAL MEMBER (INTEGER)
                 0 = ON THE NODE (INTEGER)
      3RD VALUE = LOAD MAGNITUDE (REAL)
      4TH VALUE = DISTANCE FROM THE REFERENCE
                  NODE (REAL)(IF THE 2ND VALUE = 0
                  THEN THE 4TH MUST = 0)
ENTER ALL ZEROS WHEN COMPLETED WITH POINT LOADS

[0,0,0.,0.]
  ENTER THE TYPE OF LOAD TO ADD
       1 = POINT LOADS
       2 = LINEAR LINE LOAD ALONG A MEMBER
       0 = NO MORE LOADS TO ADD
[0]
THIS SUBROUTINE ALLOWS PLASTIC MOMENTS TO BE
FORMED AT LOCATIONS OFF OF THE NODES OF THE
GRILLAGE.  THIS IS DONE BY WRITING ADDITIONAL
CONSTRAINT EQUATIONS FOR THE LINEAR PROGRAMMING
ROUTINE
AUTO CONSTRAINTS WILL LOOK AT EACH MEMBER AND
CONSTRAIN THE MAXIMUM MOMENT TO <= YIELD MOMENT


 MANUAL ALLOWS THE USER TO CHOOSE WHICH MEMBERS
 TO SET CONSTRAINTS FOR AND THEIR LOCATION


INPUT A  1  FOR AUTOMATIC CONSTRAINTS
      A  2  FOR MANUAL CONSTRAINTS
      A  0  FOR NO CONSTRAINTS

[2]
CONSTRAINED MEMBERS MUST HAVE AT LEAST ONE LOAD
ASSOCIATED WITH THEM
INPUT THE REFERENCE NODE FOR THE ADDITIONAL
CONSTRAINT, AND A 1 FOR HORIZONTAL MEMBER
OR A 2 FOR VERTICAL MEMBER (INTEGERS)
ENTER ZEROS FOR NO CONSTRAINTS

[6,1]
INPUT THE DISTANCE FROM THE REFERENCE NODE
TO LOCATE THE CONSTRAINT(REAL)

[7.5]
ENTER A 1 TO INDICATE THAT THE MEMBER HAS A
FULLY DEVELOPED YIELD MOMENT ALREADY DEVELOPED
NOTE: THIS TYPE MUST BE ENTERED LAST
ENTER A 0 IF THE HINGE IS NOT ALREADY FORMED
(INTEGERS)

[0]
```

At this point the program will run. All of the output is written to a data file that can either be written to the terminal or printed on the system hardcopy device.

GRID CHARACTERISTICS AND LOADING

NUMBER OF NODES HORIZONTAL = 4 HORIZONTAL LENGTH = 15.00
NUMBER OF NODES VERTICAL = 4 VERTICAL LENGTH = 15.00

BOUNDARY CONDITIONS

TOP	SIMPLE SUPPORTED
BOTTOM	SIMPLE SUPPORTED
LEFT	SIMPLE SUPPORTED
RIGHT	SIMPLE SUPPORTED

HORZ BEAM PLASTIC BENDING MOMENT = 0.10E+04

VERT BEAM PLASTIC BENDING MOMENT = 0.10E+04

ZX3LP ERROR NUMBER = 0

POINT LOADS ADDED TO THE GRID

REF	NODE	DIR FM NODE	DISTANCE FROM NODE	MAGNITUDE	LOA

NONE USED

LINE LOADS

BETWEEN NODES	REF NODE MAG	OTHER NODE MAG	LINE LOAD NU
6 AND 7	1.00	2.00	1
6 AND 10	1.00	2.00	2
7 AND 11	2.00	1.00	3
10 AND 11	2.00	1.00	4

NODE LOADS USED IN CALCULATIONS

NODE NUMBER	NODE LOAD
1	0.00
2	0.00
3	0.00

Figure 12 Sample Output

```
              4           0.00
              5           0.00
              6          20.00
              7          25.00
              8           0.00
              9           0.00
             10          25.00
             11          20.00
             12           0.00
             13           0.00
             14           0.00
             15           0.00
             16           0.00

  ADDITIONAL CONSTRAINTS USED
      BETWEEN NODES       TYPE USED
        0 AND   0           NONE

 **** ITERATION NUMBER IS   1 *****
 NODAL MOMENTS
      NODE NUMBER        HORIZ MNT        VERT MNT
          1               0.00E+00         0.00E+00
          2               0.00E+00         0.00E+00
          3               0.00E+00         0.00E+00
          4               0.00E+00         0.00E+00
          5               0.00E+00         0.00E+00
          6               0.10E+04         0.86E+03
          7               0.10E+04         0.10E+04
          8               0.00E+00         0.00E+00
          9               0.00E+00         0.00E+00
         10               0.10E+04         0.10E+04
         11               0.10E+04         0.86E+03
         12               0.00E+00         0.00E+00
         13               0.00E+00         0.00E+00
         14               0.00E+00         0.00E+00
         15               0.00E+00         0.00E+00
         16               0.00E+00         0.00E+00

      THE GRID LOAD FACTOR =      5.71
MAXIMUM MOMENTS AND LOCATION  ON LOADED MEMBERS

 BETWEEN NODES          MAGNITUDE       LOCATION
 HORIZONTAL MEMBERS
   6 AND   7            1242.              7.905        **
  10 AND  11            1242.              7.080        **
 VERTICAL MEMBERS
   6 AND  10            1179.              8.985        **
   7 AND  11            1179.              6.015        **
** INDICATES > YIELD MOMENT
```

Figure 12 Sample Output (continued)

CHAPTER 3

DISCUSSION and EXAMPLE PROBLEM

Work done by ABBOTT (2) resulted in a program capable of estimating grid load factors based only on nodal loads. These estimates turn out to be overly conservative. For any symmetrically loaded grillage one can expect the failure mode to also be symmetric. In applying linear programming to find a failure mode (or any technique that assumes nodal loading) a load factor and a pattern of plastic yield hinges at the grid intersections is developed. It should be noted that the solution found will not necessarily be unique.

In maximizing the number of plastic hinges for any given system there is bound to be adjacent nodes with fully developed plastic hinges. This will be true in all cases except the nine node grid. If a member has two fully developed yield moments of the same sign at each end, then the moment developed within the span must be greater than at the ends for any consistent loading on the member. The implication is clear. For any consistent loading scheme there must be additional yield moments formed at locations other that at the nodes. In comparing the results of the program GRIDS to the results presented by Hodge (1) or Abbott (2) it can be seen that a program that allows for yield moments to be formed with in a span returns a much more accurate result.

In taking a simple symmetric case and analyzing it for

various conditions the use of GRIDS can be demonstrated. The program's limits and a straight forward verification also result from this exercise. A nine node grillage was analyzed using a point load to compare relative load carrying capability and collapse modes for various boundary conditions and load conditions.

The first geometry considered is the nine node grillage with clamped boundaries. The load was first applied at the node and then gradually moved toward the edge along one member. First, if this grillage is analyzed in the same way as the simple supported grid in chapter two was, the grid load factor is found to be 8. A yield moment forms at the intersection of the members in both the horizontal and vertical directions. With the edges clamped hinges will also form at the boundary end of each beam. Setting the work done in deflecting the node to the sum of the work done in forming each of the plastic hinges will give the upper plastic limit or collapse load of the grid. The following equalities are obtained. Here F is the collapse load.

$$FL\theta = 8\ Mo\theta \qquad (14)$$

$$F = 8Mo/L \qquad (15)$$

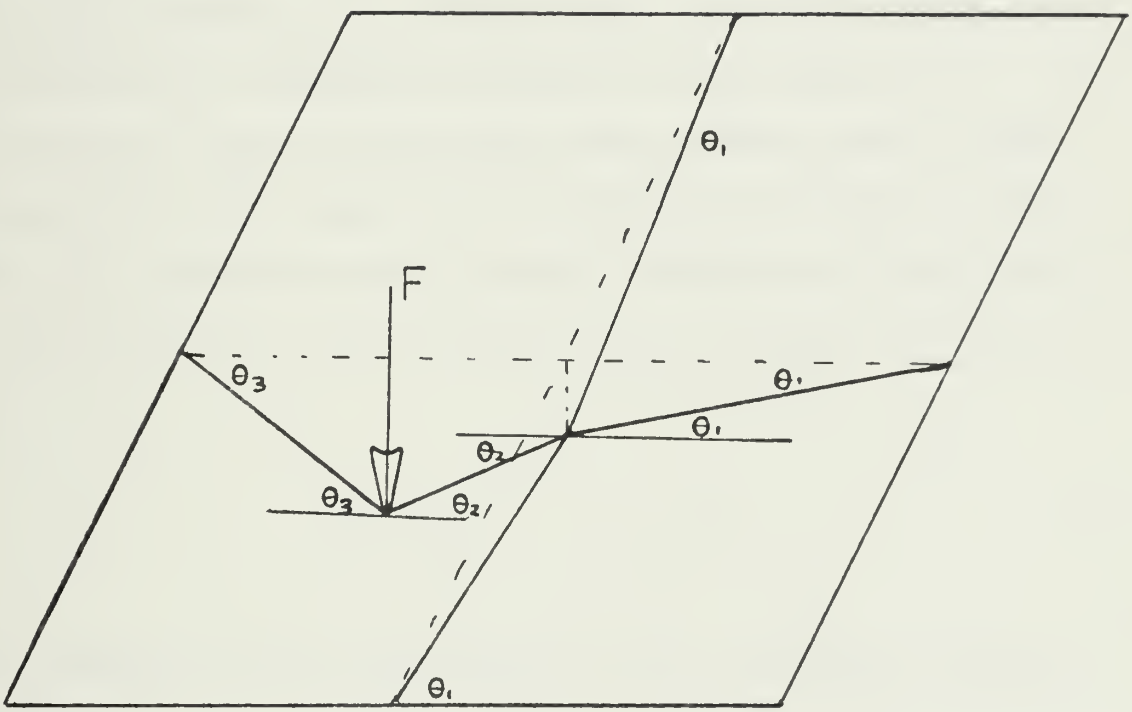


Figure 13 Collapse Mode With Plastic Hinge at Midspan

Now, if the point load is moved to midspan, we can calculate a new collapse load and allow for a plastic hinge to form at the point on the member that will see the maximum moment. Figure 14 shows the collapse mode and load configuration for a point load located half way between the node and the edge of the left center horizontal member. Because of the symmetry involved in this case it is relatively simple to calculate the collapse load by hand. Summing the total work done in deforming the grid we have the force applied moving through a distance of $\theta_3 L/2$ and the center node with its effective force displaced a distance of θ_1.

$$F/2\ \theta_3 L \quad + \quad F\ \theta_1\ L/2 \tag{16}$$

And summing the work required to form the plastic moments gives:

$$2Mo\theta_3 \quad + \quad 2Mo\theta_2 \quad + \quad 6Mo\theta_1 \tag{17}$$

Setting (3) and (4) equal to each other and realizing that θ3 is the sum of θ_1 and θ_2 gives:

$$FL(2\theta_1 + \theta_2) \quad = \quad 8Mo\ (2\theta_1 + \theta_2) \tag{18}$$

$$F \quad = \quad 8Mo/L \tag{19}$$

The fact that this result is the same as found in (15) above is only coincidental and is due to the symmetry involved. Similar calculations can be performed for any position along a member. A tabular comparison of the results of the program Grids and the results of a program that allows only nodal loading as a function of location of the point load is presented in Table 3. For the application of a point load and no intermediate yield moment, it is readily apparent that the farther the load moves from the node the more error is introduced in computing the collapse load. This occurs because the yield moment that must form at the point load will be formed with much less load as the load moves away from the node.

CLAMPED NINE NODE LOAD FACTORS

LOCATION * IN % OF LENGTH	LOAD FACTOR WITH PLASTIC MOMENTS IN THE MEMBER	LOAD FACTOR WITHOUT PLASTIC MOMENTS IN THE MEMBER
10	767.68	888.89
30	747.25	1142.86
50	800.	1600.
70	952.38	2666.67
90	2222.	8000.

* measured from the center node

Table 3 Load Factor for Various Load Locations For Plastic Moments in a Member and For no Plastic Moments Except at nodes

If the load is normalized to a total nodal load of one for each position along the member, the relative load carrying capability of the grillage can be plotted as a ratio of the load factor at a point divided by the load factor for nodal loads only versus position along a member. Table 4 lists the data for the nine node example and Figure 14 shows the result graphically.

YIELD HINGE IN MEMBER $\frac{PP}{PP_n}$	LOCATION (%) FROM CENTER NODE	NO YIELD HINGE IN MEMBER $\frac{PP}{PP_n}$	LOCATION (%) FROM CENTER NODE
1	0	1	0
0.8636	10	1	10
0.6538	30	1	30
0.5	50	1	50
0.3571	70	1	70
0.2778	90	1	90

TABLE 4 Normalized Load Factor VS Position of Hinge

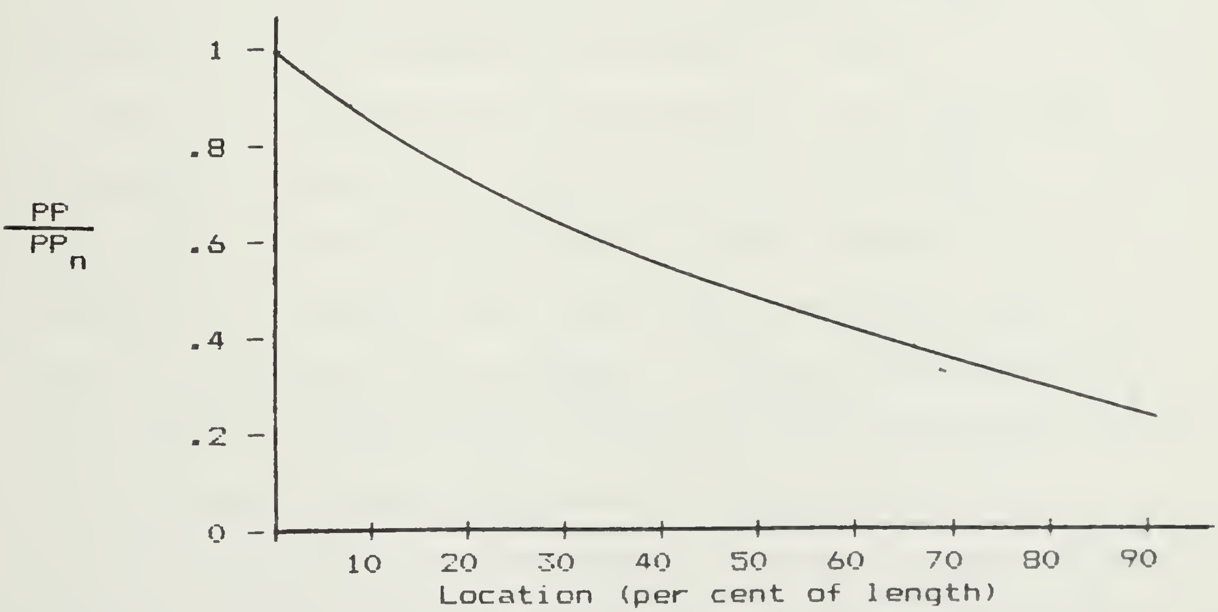


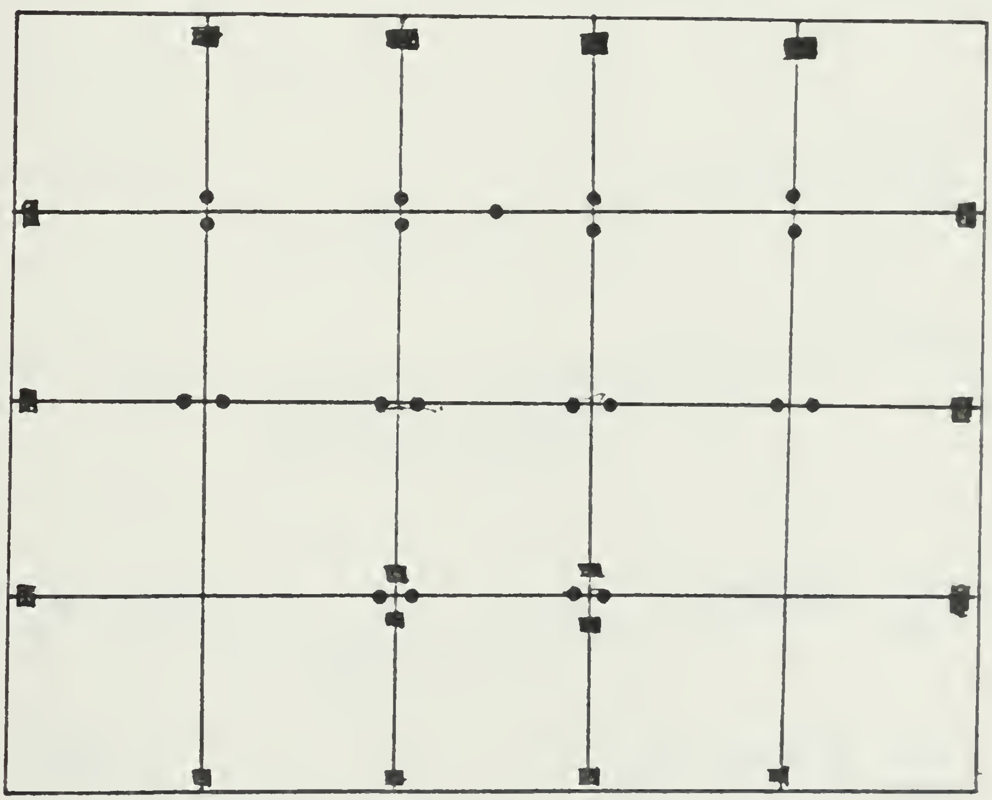
Figure 14 Normalized Collapse Load VS Position of Hinge

The second type of problem tested was a grillage that is a more realistic model of a ship's side. A 6 X 5 grillage with a line load was analyzed. A line load is a reasonable approximation of a ridge of ice along a ship's side. The line load was placed along a series of horizontal members and the load factor and failure mode were calculated. The load was moved vertically downward and the calculations were repeated for different positions along the vertical beam. Note that once the line load is off of the horizontal member, it acts like a series of point loads along a line across each of the vertical members. Figure 15 shows the loading and a typical failure mode encountered. Table 5 shows the comparison of the load factors given by grids to those that are the result of a program that allows no intermediate hinges to form. It is interesting to note that the load factor remains more nearly constant when intermediate yield moments are allowed to form.

The final problem to be considered is one that was considered by Abbott (2). Here we must convert the distributive load into line loads on the members. The problem is an example that illustrates how GRIDS might be used in considering the strength of a ship's side that is anticipated to operate in a particular ice environment. For this problem consider that the ship is to operate in a uniform ice sheet five feet thick. Further assume that the ice has a local crushing strength of 300 psi and that

LOCATION (%)	WITH INTERMEDIATE PLASTIC MOMENTS	NO INTERMEDIATE PLASTIC MOMENTS
0	384.00	400.00
10	380.36	413.79
30	353.28	444.44
50	337.78	444.44
70	330.78	392.16
90	330.83	350.88

Table 5 Load Factor for 5 X 6 VS Position of Line Load



• = POSITIVE YIELD MOMENT

■ = NEGATIVE

Figure 15 Failure Mode for Line Load on Horizontal Member

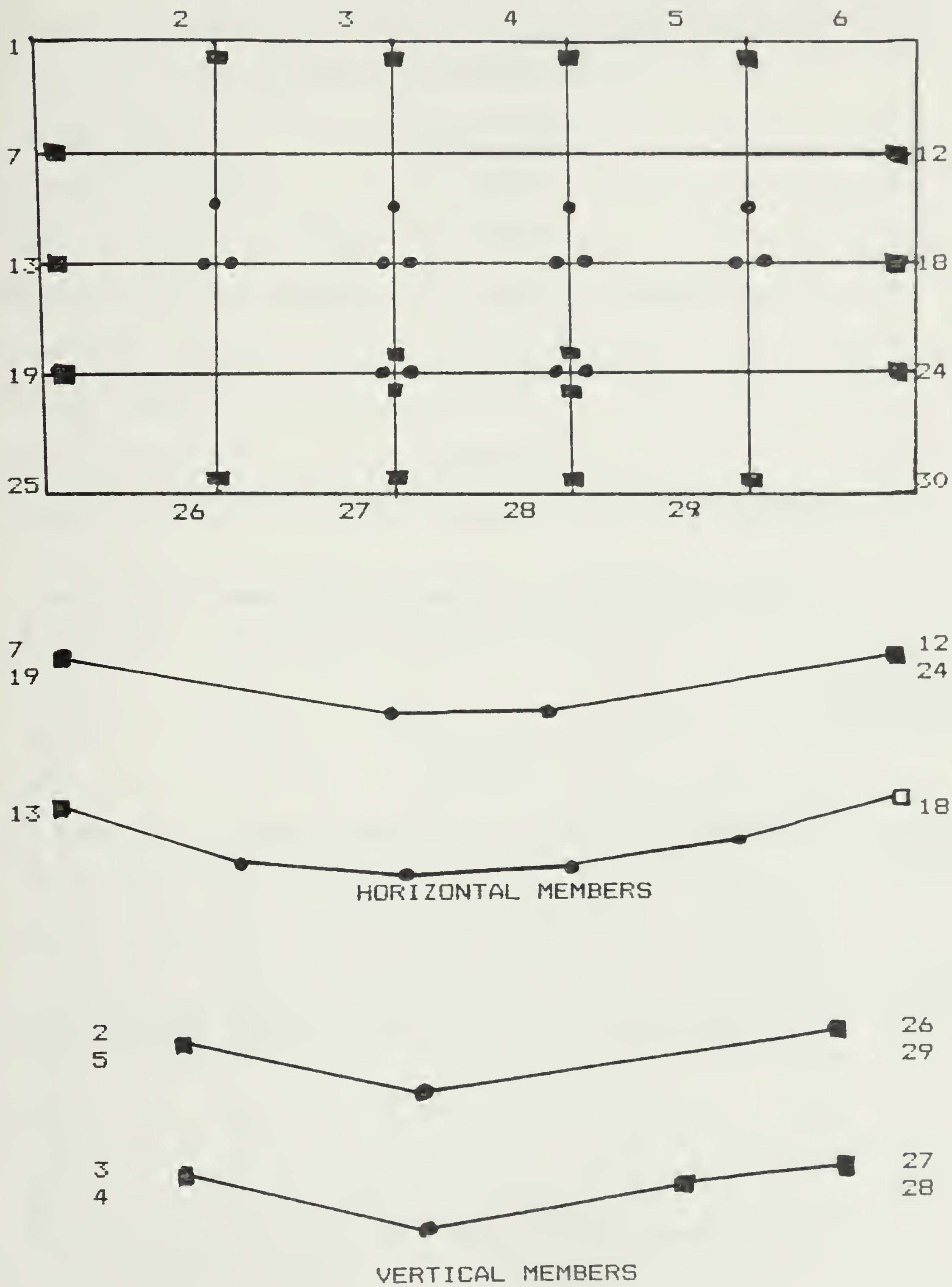


Figure 16 Failure Mode and Deformation Profile for 6 X 5 with a Horizontal Line Load Between Horizontal Members With Clamped Boundaries

crushing is the predominant failure mode. Analyze a ship which has the following characteristics.

Frame Spacing	21 inches
Bulkhead Spacing	7 feet
Distance Between Decks	15 feet
Number of longitudinals	2 (evenly spaced)
Stringer Modulus	.5 in^3
Frame Modulus	.8 in^3
Yield strength	50000 psi

Figure 17 shows the configuration for this problem.

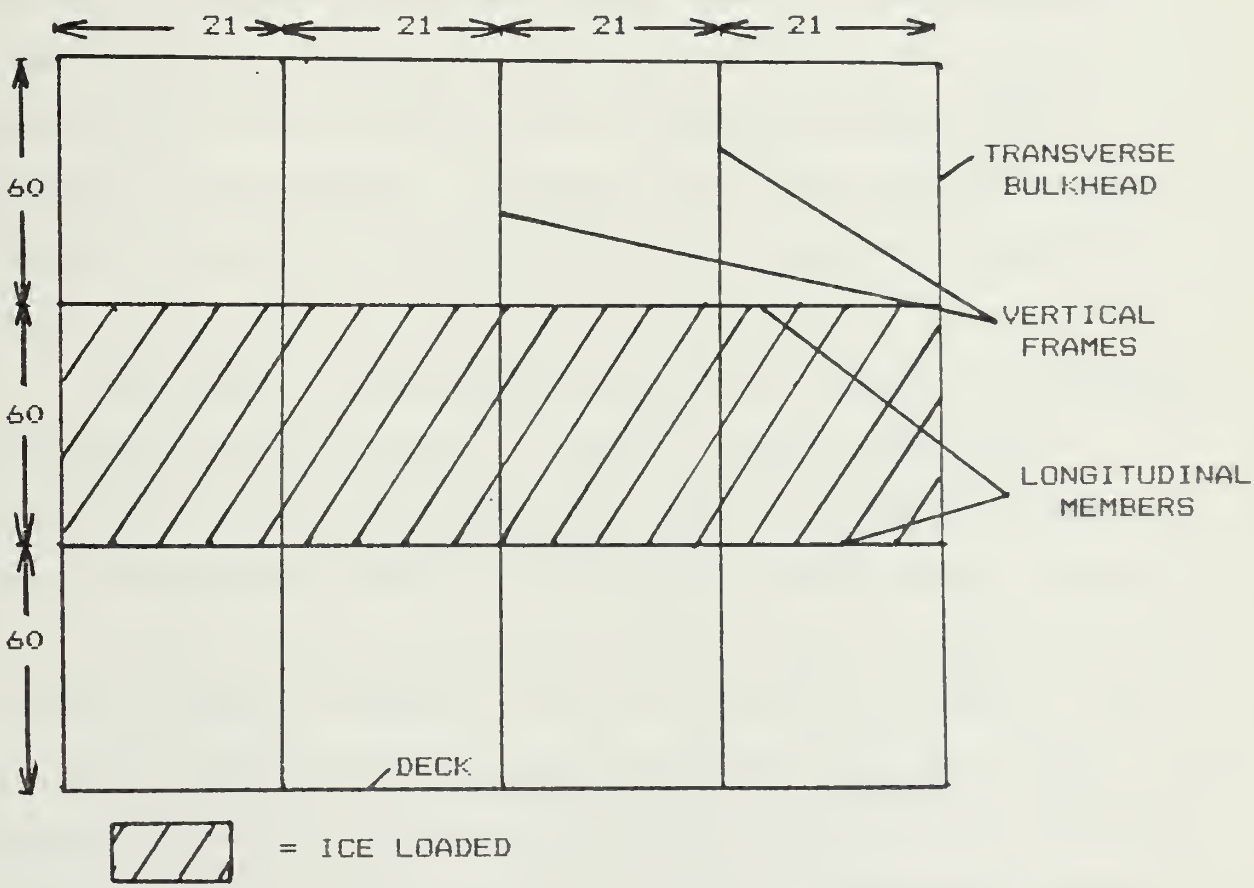


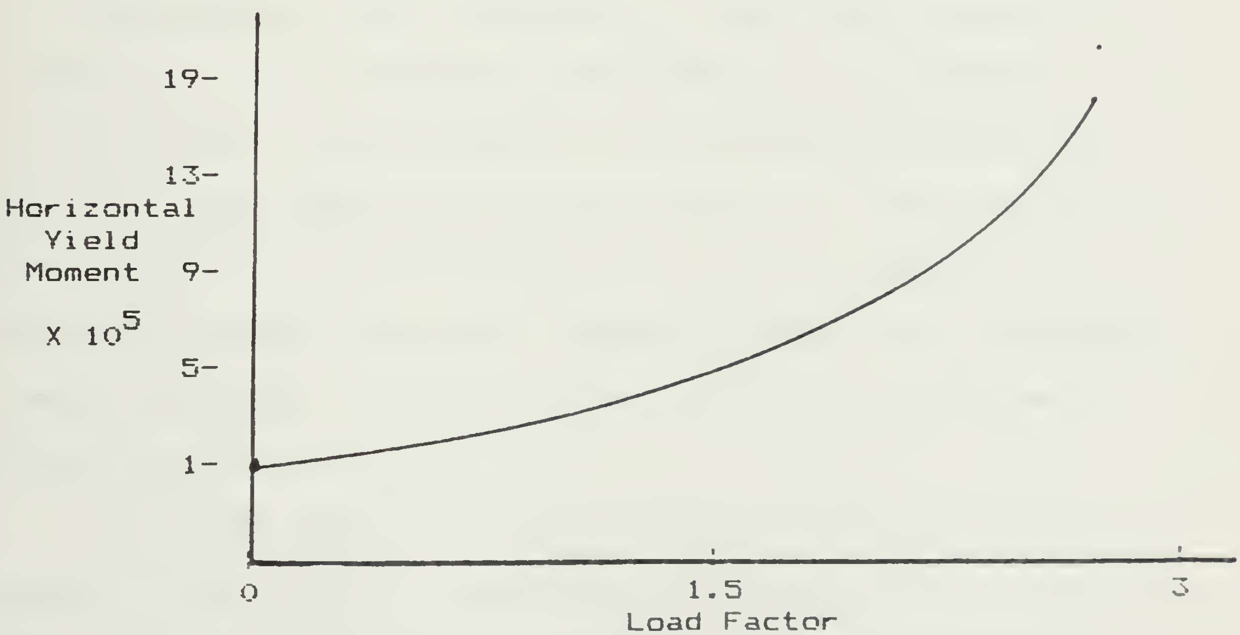
Figure 17 Ice Loaded Ship's Side

Converting the distributive ice loads on this grillage we get lineloads around each plate element. Each plate has a total pressure of 3.78×10^5 pounds on it. Assuming this reduces to a uniform line load around the perimeter of each plate results in a line load of 583.33 pounds per inch. Entering the data and selecting automatic constraints results gives a Load Factor of 0.175. With this load factor we know that this grillage can support less than twenty percent of the anticipated load. The structure must be strengthened. Strengthening can be accomplished in several ways. Increasing plate thickness, frame of longitudinal member sizes or using stronger materials are all viable options that will result in an increased maximum yield moment of each member. Increasing the yield moment of each member results in a grillage that will support the applied load.

Additional insight can ge gained by varying the strength of either the horizontal or vertical members alone and observe the structural behavior. With the same load applied as in the problem above, the horizontal yield moment was set at a value of 100,000 and the vertical bending moment was varied through a range of values of 20,000 to 90,000. All of these values give the same Load Factor of 0.18. This suggests that for the applied loading the grillage is much weaker in the horizontal direction, which makes sense when the geometry is considered. In looking at the failure mode

provided by GRIDS we observe that plastic hinges are indicated for all boundary nodes. We also know that there are a number of yield moments given that do not actually occur, due to the degree of indeterminancy and linear programming methods (Chapter 4 contains additional discussion on this point). We can conclude from the above that the moments in the vertical members are the ones that are not actually formed.

Now, if we hold the yield moment of the vertical member constant and vary the yield moment of the horizontal member we can observe the grillage behavior as a function of the horizontal member size. Figure 16 shows how the load factor varies as the horizontal member yield moment for this example.



Vertical Yield Moment = 90,000 in^4

Figure 18 Load Factor Versus Horizontal Member Strength

CHAPTER 4

SUMMARY OF RESULTS

As illustrated in Chapter 3 the results that Grids gives are significantly more accurate than for a program that allows only nodal loading. In choosing to use linear programming to solve a problem with intermediate yield hinges, an iterative approach is also required. If one were to write constraint equations for forming the plastic hinges within a member directly, as a function of a general combination of point and line loads, the resulting equations would be non-linear. In order to use linear programming and still allow plastic hinges in a member, the end moments must be used. So, the problem is first solved using only nodal loads to calculate the end moments. Constraint equations are then written that are a linear combination of loading and end moments. The assumption is that the final end moments will not change the basic shape of the moment distribution. The end moment only changes the slope of the line that the moment distribution lies on. From this one can see that if the end moments, as given by GRIDS, for a particular member are equal then the location of the maximum moment is exact. Hence the position of the yield moment in that member is correct.

For point loads only the position of the failure is known and the program solution is accurate. In the case of line loads GRIDS gives a "linearized" approximation.

Calculating the approximate error can be accomplished by comparing the location of the maximum moment based on the moment distribution and member end moments to the position that the additional constraint was written for. Doing this assumes that the yield moment that forms at the location of the constraint does not change the moment distribution. For a constant line load, the location error in percent of the length of the member is approximately one half of the percent difference between end moments. In other words, if the end moments are within twenty percent of each other then the plastic yield moment will be within ten percent of the location given by GRIDS. Further more, it will be shifted toward the end of the member with the larger moment.

The location that GRIDS calculates for the maximum will be exact for point loads and constant line loads on a sysmmetrical grid with symmetrical loading. In non-symmetrical grids with multiple load types the locations of the maximum moments are approximate but very close to actual. This occurs because of the way in which linear programming calculates the end moments of each member. If the end moments of a member are unchanged between the first and second iterations as labeled in the output, then the location of the maximum moment in that member calculated by grids is exact. If one of the end moments of the member is reduced then the effect is to lower that end of the moment curve until the maximum moment as constrained is found. The

curve keeps its original shape however. The error introduced is small as long as the end point magnitudes of a member's linear line loads do not differ more than fifty percent.

GRIDS offers both automatic and manual modes for developing the additional constraint equations that actually add the plastic yield moments in a member. This gives added flexibility in evaluating a particular grillage. By choosing the manual mode and choosing equality constraints a failure can be force to form at a particular location. By forcing a failure, or by having an already formed plastic hinge in a member, the load carrying capability of a partially failed grid can be examined. This technique may be desirable to evaluate a grillage that has undergone a local impact on a member so that a hinge may be formed at a location first that might otherwise not reach its yield moment until much higher general loads are applied. If, however, a plastic hinge is placed in a member with no loads associated with it the output is meaningless and a ZX3LP linear programming error will occur.

As discussed previously, the failure mode provided by GRIDS is not unique. All of the systems encountered contain more unknowns than equations and therefore cannot be solved directly. The way in which this linear system is solved and maximized causes selected inequality constraints to be evaluated as equality constraints. The number used in this

fashion is equal to the degree of indeterminancy of the system. This means that not all of the locations that are listed in the output may actually reach the yield moment. But, the output provided is a maximized and feasible solution. In other words, there may be fewer yield moments formed in an actual failure but the total work done is correct. This could be accounted for by different magnitudes of rotation of the hinges that do form.

FUTURE WORK

GIRDS can be modified to handle a much broader base of problem than those which have been considered up to now. There are some modifications and additions that could be made to enhance the program's performance.

The most useful feature that could be added would be the capability to enter distributed loads. This task should not be too simplistic, however. Distributed plate loads do not transfer to the edges of the plate as linear line loads except in the case of circular plates and should not be handled in this way. One might approach this problem by linearizing or approximating the Fourier series solution for loaded plates.

Another area that improvement could be made is to allow for different yield moments in particular members. Adding arrays to keep track of this specific information and calling new variables as appropriate would be tedious but not particularly difficult from a programming standpoint.

REFERENCES

1. Tait, P. and P.G. Hodge, "BEAMPLT: A Program for finding the Yield Point Load of Transversely Loaded Rectangular Grids", Office of Naval Research Report No. AFM-H1-26, Chicago, Illinois, January, 1981.

2. Abbott, G.1., "Strengthening Criteria for Grillages Subjected to Ice Loads", Ths. Massachusetts Institute of Technology, 1982

3. Glenn, L.E. and H. Blount, "Measurement of Ice Impact Pressure and Loads On Board CCGS Louis St. Laurent", Third International Offshore Mechanical and Arctic Engineering Symposium, Volume III, American Society of Mechanical Engineers, February, 1984.

4. St. John, J.W. and C. Daley, "Shipboard Measurement of Ice Pressures in the Bering, Chukchi, and Beaufort Seas", Third International Offshore Mechanical and Arctic Engineering Symposium, Volume III, American Society of Mechanical Engineers, 1984.

5. Heyman, J., "The Limit Design of a Transversely Loaded Square Grid", Journal of Applied Mechanics, ASME, June, 1952.

6. Johansson, B.M., "On the Ice-Strengthening of Ships Hulls", International Shipbuilding Progress; Shipbuilding and Marine Monthly, Vol.14, No. 154,1967.

7. Hodge, P.G., Plastic Analysis of Structures, McGraw-Hill Book Co., New York, 1964.

8. Glass, S.j., Linear Programming, Second Edition, McGraw-Hill Book Co., New York, 1964.

9. Hadley, G., Linear Programming, Addison-Wesley, Reading, Massachusetts, 1962.

10. Murer, C., "Strengthening of Hull Structures in Ice", Diss., Seventh Wegemt School, Helsinki, March, 1983.

APPENDIX A

PROGRAM LISTING AND DIRECTIONS

Grids is a very simple program to use. In the interactive mode it is self explanitory and requires no further information to run. To run the program more quickly when repetitive inputs are desired, an input data file may be used. Figure 19 shows the construction of such a file. One need only assign that file to be the source of input to run the program with all input read from the input file without further action from the user.

The Grillage size limits placed on GRIDS was primarily for the convinence of the programmer. Arrays may be redimensioned to solve larger grids. The criteria provided in Appendix B for the IMSL linear programming routine must be met,however.

SAMPLE INPUT DATA FILE (WITH EXPANATIONS)

```
5,4                           GRID DIMENSIONS ON THIS LINE

2,2,2,2                       BOUNDARY CONDITIONS

21.                           HORIONTAL LENGTH

60.                           VERTICAL LENGTH

30000.                        HORIZONTAL YIELD MOMENT

90000.                        VERTICAL YIELD MOMENT

2                             TYPE OF LOAD (1,2,0)

6,1,53.3,53.3\
6,2,53.3,53.3  \
7,2,53.3,53.3    \
7,1,53.3,53.3      \
8,1,53.3,53.3                 REFERENCE NODE, DIRECTION (1,2),
8,2,53.3,53.3
9,2,53.3,53.3                 VALUE FOR POINT LOAD, OR A-NODE
9,1,53.3,53.3
10,2,53.3,53.3                VALUE FOR LINE LOAD, LOCATION FOR
11,1,53.3,53.3
12,1,53.3,53.3     /          POINT LOAD, OR B-NODE VALUE FOR LINE
13,1,53.3,53.3  /
14,1,53.3,53.3/

0,0,0.,0.                     NO MORE LOADS OF THIS TYPE TO ADD

0                             NO MORE LOADS TO ADD

1                             TYPE OF CONSTRAINTS (1,2,0)
```

There may be no blank lines in the data file. Lines were left blank for clarification. There are four additional lines required if manual constraints are used.

Figure 19 Sample Input Data File

```
C     THIS BLOCK IS THE COMMON MEMORY SHARED BY ALL SUBROUTINES
C     IT IS INCORPORATED BY USING THE INCLUDE ATATEMENT

        COMMON /A/ PTLOAD(30),NODTOT,NODESH,NODESV,
     +  II(30),JJ(30),VLEN,HLEN,XDCNST(30),
     +  LD1ND(60),D1VAL(60),D1LOC(60),LPT,LD2ND(60),
     +  D2AVL(60),D2BVL(60),LNLDN,LDIS,NDCNST(30),
     +  LD3ND(60),D3VAL(60),IAUTO,NEQST

        COMMON /B/ VMNT(30),HMNT(30),HORZMO,VERTMO
        COMMON /C/ ITOP,ILFT,IRGT,IBTM,ITERAT
        COMMON /D/ PP,IER,IEQNM,NODEEQ,NITRAT
        COMMON /E/ FOG(12,61),A2(31,61),B1(12),B2(31)
        COMMON /F/ LIN(40),MN,IA,LMAX,M1,M2,IW1,IRW1,LP2

        COMMON /GGG/ BHKMNT(30),BHKLOC(30),BVKMNT(30),BVKLOC(30)

C      THIS PROGRAM SOLVES THE PROBLEM OF FINDING THE MAXIMUM LOAD FACTOR
C      TO BE APPLIED TO A PARTICULAR LOADING CONDITION OF A RECTANGULAR GR
C      IT SOLVES THE PROBLEM BY CONVERTING THE RESULTING SET OF EQUATIONS
C      INTO A LINEAR PROGRAMMING PROBLEM.  AT THE PRESENT TIME THE MAXIMUM
C      SIZE OF GRID THAT CAN BE HANDLED IS A 6 X 5 GRID (IE 30 NODES )
C      THIS CONTRAINT IS ARTIFICIAL AND IS DUE SOLELY TO THE VERY LARGE
C      WORK VECTOR REQUIRED BY THE LP SUBROUTINE ZX3LP
C      SEE IMSL LIBRARY REFERENCE MANUAL #8 FOR FURTHER INFO.

        DIMENSION  A(83,39),B(83),C(39),PSOL(83),DSOL(83),
     +  RW(7200),IW(240)


        FOR'



       OPEN(UNIT=15,FILE='MOMENTS.DAT',STATUS='NEW')
1             PRINT*,'PROGRAM RESTRICTED TO A MAXIMUM GRID OF 6 X 5'
              PRINT*,'IE 30 NODES TOTAL OR LESS'
              PRINT*,'THIS RESTRICTION IS IMPOSED BECAUSE OF THE MAXIMUM'
              PRINT*,'SIZE OF THE WORK VECTOR USED BY ZX3LP IS CONSTRAINED'
              PRINT*,'BY THE PROGRAMMMER.'
              PRINT*,'SEE PROGRAM MANUAL FOR MORE INFO'
C
C             INPUT SIZE OF GRID TO BE SOLVED
              PRINT*,'     '
              PRINT*,'     '
C
       PRINT*,'INPUT NUMBER OF NODES ALONG THE TOP AND SIDE OF THE GRID'
       READ(5,*) NODESH,NODESV
              NODTOT=NODESH*NODESV
              IF(NODTOT.GT.30) GOTO 1
C
C             INPUTCONDITIONS
```

```
C
2     PRINT*,'INPUT BOUNDARY CONDITIONS FOR THE TOP'
      PRINT*,'LEFT,RIGHT AND BOTTOM EDGES OF THE GRID'
      PRINT*,' IN THAT ORDER'
      PRINT*,'INPUT A 1 FOR SIMPLY SUPORTED EDGE'
      PRINT*,'INPUT A 2 FOR A CLAMPED EDGE'
      PRINT*,'ALL VALUES ARE INTEGERS'
           PRINT*,'     '
           PRINT*,'     '
           READ(5,*) ITOP,ILFT,IRGT,IBTM
           I=ITOP
           J=ILFT
           K=IRGT
           L=IBTM
      IF(I.EQ.1.OR.I.EQ.2.AND.J.EQ.1.OR.J.EQ.2.
     +AND.K.EQ.1.OR.K.EQ.2.AND.L.EQ.1.OR.L.
      PRINT*,'BOUNDARY CONDITIONS NOT ENTERED CORRECTLY.'
      PRINT*,' TRY AGAIN'
      GO TO 2
3     CONTINUE
      PRINT*,'INPUT HORIZONTAL MEMBER LENGTH (REAL NUM.)'
           PRINT*,'     '
           READ(5,*) HLEN
      PRINT*,'INPUT VERTICAL MEMBER LENGHT (REAL NUM.)'
           PRINT*,'     '
           READ(5,*) VLEN
      PRINT*,'INPUT HORIZONTAL BEAM YIELD MOMENT (REAL)'
           PRINT*,'     '
           READ(5,*) HORZMO
      PRINT*,'INPUT VERTICAL BEAM YIELD MOMENT (REAL)'
           PRINT*,'     '
           READ(5,*) VERTMO
C     SET LOAD COUNTERS TO ZERO
           LPT=0
           LNLDN=0
           LDIS=0
           IEQNM=0
C
C     NOW TO INITIALILZE THE LOAD AND MOMENTS ARRAYS
C
      CALL MINT


      CALL INPTL

C     NOW TO ENTER ALL THE LOADS APPLIED TO THE GRID
C

      CALL ENTERL

C
C       NOW TO ENTER ADDITIONAL CONSTRAINT INEQUALITIES
```

```
C        SEE USER'S HANDBOOK FOR FURTHER INFO
C
         PRINT*,'THIS SUBROUTINE ALLOWS PLASTIC MOMENTS TO BE'
         PRINT*,'FORMED AT LOCATIONS OFF OF THE NODES OF THE'
         PRINT*,'GRILLAGE.  THIS IS DONE BY WRITING ADDITIONAL'
         PRINT*,'CONSTRAINT EQUATIONS FOR THE LINEAR PROGRAMMNIG'
         PRINT*,'ROUTINE'
         PRINT*,'AUTO CONSTRAINTS WILL LOOK AT EACH MEMBER AND'
         PRINT*,'CONSTRAIN THE MAXIMUM MOMENT TO <= YIELD MOMENT'
         PRINT*,'      '
         PRINT*,'      '
         PRINT*,' MANUAL ALLOWS THE USER TO CHOOSE WHICH MEMBERS'
         PRINT*,' TO SET CONSTRAINTS FOR AND THEIR LOCATION '
         PRINT*,'              '
         PRINT*,'     '
         PRINT*,'INPUT A  1  FOR AUTOMATIC CONSTRAINTS'
         PRINT*,'      A  2  FOR MANUAL CONSTRAINTS'
         PRINT*,'      A  0  FOR NO CONSTRAINTS    '

                 READ(5,*) IAUTO
                 IF(IAUTO.EQ.2) THEN
                    CALL CONST
                 ENDIF


5        CONTINUE
C
C      NOW TO CALCULATE THE NUMBER OF NODAL EQUATIONS
C
            NODEEQ=(NODESH-2)*(NODESV-2)
C
C      NOW TO CALCULATE THE NODAL EQUATION OF THE GRID AND WRITE THEM TO
C      THE MATRIX FOG
C
       CALL EQNS


C      WE MUST CALCUALTE THE DIMENSIONS OF ALL ARRAYS BEFORE
C      FIXER BECAUSE ZX3LP REQUIRES IT
C      NOW TO CALCULATE AND TRACK THE TOTAL NUMBER OF VARIABLES
C           IE TOTAL NUMBER OF NO-ZERO MOMENTS + LOAD FACTOR
C
         L=1
         DO 15,I=1,NODTOT
            IF(HMNT(I).LT.0.5) GO TO 15
            LIN(L)=I
            L=L+1
15       CONTINUE
         LP2=L-1
         DO 16 I=1,NODTOT
            IF(VMNT(I).LT.0.5) GO TO 16
            LIN(L)=I
```

```
            L=L+1
16        CONTINUE

          LMAX=L
C
C         NOW TO MASSAGE THE INFORMATION CALCULATED AND STORED BY
C         THE PROGRAM SO THAT IT CAN BE USED BY ZX3LP
C
          M1=LMAX+IEQNM-1
          M2=NODEEQ
          IA=M1+M2+2
          IRW1=IA*IA+3*M1+2*M2+4
          IW1=2*M2+3*M1+4
              MN=M1+M2
C ADJUST M1 AND M2 IF THERE ARE EQUALITY CONSTRAINTS
          M1=M1-NEQST
          M2=M2+NEQST

        CALL FIXER (A,B,C,PSOL,DSOL,RW,IW)
C CALL BEAM CHECKING ROUTINE
        CALL BEAMCK
          ITERAT=ITERAT+1
C CALL OUTPUT
             CALL OUTPUT

          BMAX=0.0
        DO  999 I=1,NODTOT
          IF(BHKMNT(I).GT.HORZMO)THEN
             BMAX=BHKMNT(I)
          ENDIF
          IF(BVKMNT(I).GT.VERTMO)THEN
             BMAX=BVKMNT(I)
          ENDIF
999       CONTINUE
C LOOP TO WRITE EXTRA CONSTRAINT EQUATIONS IF AUTOMATIC IS SELECTED
          IF(IAUTO.EQ.1.AND.BMAX.GT.0.0.AND.
     +          ITERAT.LT.2) THEN
             CALL CONST
             GO TO 5
          ENDIF

          END




          SUBROUTINE   ENTERL

          INCLUDE 'COMMON.FOR'

          PRINT *,'  LOADS ARE ENTERED REFERENCED TO THE NODE'
```

```
      PRINT *,'  ABOVE OR TO THE LEFT OF THE LOAD'
      PRINT *,'  ALLOWED LOADING CONFIGURATIONS:'
      PRINT *,'      1. POINT LOADS'
      PRINT *,'      2. LINEAR LINE LOADS ALONG A MEMBER'
      PRINT *,'      3. LOAD ON A PLATE ELEMENT'
      PRINT *,'           '


      PRINT *,'         '
      PRINT *,'         '
 90   PRINT *,'  ENTER THE TYPE OF LOAD TO ADD'
      PRINT *,'         1 = POINT LOADS'
      PRINT *,'         2 = LINEAR LINE LOAD ALONG A MEMBER'
      PRINT *,'         0 = NO MORE LOADS TO ADD'

      READ(5,*)MM

      IF (MM .EQ. 0) GO TO 150
      IF (MM .EQ. 1) GO TO 100
      IF (MM .EQ. 2) GO TO 110

100   CALL PNTLD (PTLOAD,NODTOT,NODESH,NODESV,VLEN
     + ,HLEN,II,JJ,LD1ND,D1VAL,D1LOC,LPT)
      GO TO 90
110   CALL LNLD (PTLOAD,NODTOT,NODESH,NODESV,VLEN
     + ,HLEN,II,JJ,LD2ND,D2AVL,D2BVL,LNLDN)
      GO TO 90

150   RETURN
      END




      SUBROUTINE    PNTLD (PTLOAD,NODTOT,NODESH,NODESV,VLEN
     +                    ,HLEN,II,JJ,LD1ND,D1VAL,D1LOC,LPT)

     DIMENSION PTLOAD(NODTOT),D1LOC(NODTOT*2)
     DIMENSION D1VAL(NODTOT*2)
     DIMENSION II(NODTOT),JJ(NODTOT),LD1ND(NODTOT*2)


     PRINT *,'THIS ROUTINE ADDS POINT LOADS TO THE GRID.
    + THERE MAY BE'
     PRINT *,'A MAXIMUM OF ONE POINT LOAD PER NODE PLUS ONE
    +PER MEMBER'
     PRINT *,'THE REFERENCE NODE IS ABOVE OR TO THE LEFT OF
    + THE LOAD'

5     PRINT *,'ENTER FOUR VALUES TO DEFINE EACH LOAD'
      PRINT *,'       1ST VALUE = REFERENCE NODE (INTEGER)'
      PRINT *,'       2ND VALUE = VERTICAL OR HORIZONTAL MEMBER'
```

```
      PRINT *,'                    1 = HORIZONTAL MEMBER (INTEGER)'
      PRINT *,'                    2 = VERTICAL MEMBER (INTEGER)'
      PRINT *,'                    0 = ON THE NODE (INTEGER)'
      PRINT *,'          3RD VALUE = LOAD MAGNITUDE (REAL)'
      PRINT *,'          4TH VALUE = DISTANCE FROM THE REFERENCE'
      PRINT *,'                      NODE (REAL)(IF THE 2ND VALUE = 0'
      PRINT *,'                      THEN THE 4TH MUST = 0)'

      PRINT *,'ENTER ALL ZEROS WHEN COMPLETED WITH POINT LOADS'

      READ (5,*) NDNUM,IDIR,VAL,RLOCA

      IF (NDNUM .EQ. 0) GO TO 130
      IF (NDNUM .GT. NODTOT) GO TO 100
      IF (II(NDNUM) .EQ. NODESV AND IDIR .EQ. 2) GOTO 110
      IF (JJ(NDNUM) .EQ. NODESH AND IDIR .EQ. 1) GOTO 110
      IF (IDIR .EQ. 2 .AND.RLOCA .GT. HLEN) GO TO 120
      IF (IDIR .EQ. 1 .AND. RLOCA .GT. VLEN) GO TO 120
      IF (IDIR .EQ. 2) THEN
          RLOCA = -1.0 * RLOCA
      ENDIF
      LPT = LPT + 1
C     NEGATIVE RLOCA INLICATES LOAD ON THE VERTICAL MEMBER

      LD1ND(LPT) = NDNUM
      D1VAL(LPT) = VAL
      D1LOC(LPT) = RLOCA

C     ADD LOAD IF IT IS ON THE NODE
      IF (IDIR .EQ. 0 ) THEN
          PTLOAD(NDNUM) = PTLOAD(NDNUM)+VAL
      GO TO 5
      ENDIF

C     SET DUMMY VARIABLES FOR VERT OR HORIZ MEMBER AS APPROP
      IF (IDIR .EQ. 1) THEN
          NB=NDNUM+1
          RLEN = HLEN
        ELSE
          NB=NDNUM+NODESH
          RLEN = VLEN
      ENDIF

C     CALCULATE AND ADD NODE LOADS FOR POINT LOAD APPLIED
     PTLOAD(NB) = PTLOAD(NB) + (ABS(RLOCA)*VAL/RLEN)
     PTLOAD(NDNUM) = PTLOAD(NDNUM)+(RLEN-ABS(RLOCA))*VAL/RLEN

      GO TO 5

 100  PRINT *,'THE LAST NODE ENTERED IS NOT ON THE GRID,'
      PRINT *,'RE-ENTER ALL DATA FOR THE LAST LOAD'
      GO TO 5
```

```
110     PRINT *,'THE LAST LOAD ENTERED IS NOT ON THE GRID.'
        PRINT *,'RE-ENTER ALL DATA FOR THE LAST LOAD'
        GO TO 5

120     PRINT *,'LOAD LOCATION IS NOT ON THE MEMBER ADJACENT TO'
        PRINT *,'THE REFERENCE NODE, RE-ENTER ALL DATA FOR THE'
        PRINT *,'LAST LOAD'
        GO TO 5

130     RETURN
        END


        SUBROUTINE LNLD(PTLOAD,NODTOT,NODESH,NODESV,VLEN
     + ,HLEN,II,JJ,LD2ND,D2AVL,D2BVL,LNLDN)

       DIMENSION PTLOAD(NODTOT),D2AVL(NODTOT),D2BVL(NODTOT)

       DIMENSION LD2ND(NODTOT),II(NODTOT),JJ(NODTOT)

        PRINT *,' THIS ROUTINE ADDS LINEAR LOADS. THE LOADS MUST BE
     +ON THE'
        PRINT *,' GRID MEMBER.  THE MAXIMUM NUMBER OF LINEAR LOADS
     + ALLOWED'
        PRINT *,' IS ONE PER GRID MEMBER.  THE REFERENCE NODE IS
     + ABOVE OR'
        PRINT *,' TO THE LEFT OF THE LOAD.                               '

5       PRINT *,'ENTER FOUR VALUES TO DEFINE EACH LOAD                   '
        PRINT *,'     1ST VALUE = REFERENCE NODE (INTEGER)               '
        PRINT *,'     2ND VALUE = VERTICAL OR HORIZONTAL MEMBER          '
        PRINT *,'                 1 = HORIZONTAL (INTEGER)               '
        PRINT *,'                 2 = VERTICAL (INTEGER)                 '
        PRINT *,'     3RD VALUE = MAGNITUDE AT REFERENCE NODE (REAL)'
        PRINT *,'     4TH VALUE = MAGNITUDE AT OPPOSITE NODE (REAL) '

        PRINT *,'ENTER ZEROS FOR ALL FOUR VALUES WHEN COMPLETED         '

        READ (5,*) NDNUM,IDIR,AVAL,BVAL


        IF (NDNUM .EQ. 0) GO TO 130
        IF (NDNUM .GT. NODTOT) GO TO 100
        IF (II(NDNUM) .EQ. NODESV AND IDIR .EQ. 2) GOTO 110
        IF (JJ(NDNUM) .EQ. NODESH AND IDIR .EQ. 1) GOTO 110
        IF (IDIR .EQ. 2) THEN
            BVAL = -1.0*BVAL
            AVAL=-1.0*AVAL
        ENDIF
        LNLDN = LNLDN + 1
C       NEGATIVE BVAL OR AVAL  INDICATES LOAD ON THE VERTICAL MEMBER
```

```
      LD2ND(LNLDN) = NDNUM
      D2AVL(LNLDN) = AVAL
      D2BVL(LNLDN) = BVAL

C     SET DUMMY VARIABLES FOR VERT OR HORIZ MEMBER AS APPROP
      IF (IDIR .EQ. 1) THEN
          NB=NDNUM+1
          RLEN = HLEN
        ELSE
          NB=NDNUM+NODESH
          RLEN = VLEN
      ENDIF
              BVAL=ABS(BVAL)
              AVAL=ABS(AVAL)
      IF (AVAL .GT. BVAL) THEN
         AA = 6.0
         BB = 3.0
         W = BVAL
         P = AVAL - BVAL
      ELSE
         AA = 3.0
         BB = 6.0
         W = AVAL
         P = BVAL - AVAL
      ENDIF

C     CALCULATE AND ADD NODE LOADS FOR LINE LOAD APPLIED

      PTLOAD(NDNUM) = PTLOAD(NDNUM) +((W*RLEN/2)+P*RLEN/BB)
      PTLOAD(NB) = PTLOAD(NB) + ((W*RLEN/2) + P*RLEN/AA)

      GO TO 5

 100  PRINT *,'THE LAST NODE ENTERED IS NOT ON THE GRID,'
      PRINT *,'RE-ENTER ALL DATA FOR THE LAST LOAD'
      GO TO 5

 110  PRINT *,'THE LAST LOAD ENTERED IS NOT ON THE GRID,'
      PRINT *,'RE-ENTER ALL DATA FOR THE LAST LOAD'
      GO TO 5

 130  RETURN
      END

    SUBROUTINE MINT


      INCLUDE 'COMMON.FOR'
```

```
C
C     SUBROUTINE TO INITIALIZE THE BENDING MOMENT ARRAYS BASED ON THE
C     GIVEN BOUNDARY CONDITIONS
C
      K=0
      DO 11  I=1,NODESV
           DO 10 J=1,NODESH
           NDNUM=K*NODESH + J
           IF(ITOP.EQ.1.AND.I.EQ.1) GO TO 100
           IF(ITOP.EQ.2.AND.I.EQ.1) GO TO 150
           IF(IBTM.EQ.1.AND.I.EQ.NODESV) GO TO 100
           IF(IBTM.EQ.2.AND.I.EQ.NODESV) GO TO 150
           IF(ILFT.EQ.1.AND.J.EQ.1) GO TO 100
           IF(ILFT.EQ.2.AND.J.EQ.1) GO TO 300
           IF(IRGT.EQ.1.AND.J.EQ.NODESH) GO TO 100
           IF(IRGT.EQ.2.AND.J.EQ.NODESH) GO TO 300
      VMNT(NDNUM)=1.0
      HMNT(NDNUM)=1.0
      II(NDNUM)=I
      JJ(NDNUM)=J
      GO TO 10
150     IF(J.EQ.1.OR.J.EQ.NODESH) THEN
                GO TO 100
        ELSE
                GO TO 200
        ENDIF
100        VMNT(NDNUM)=0.0
           HMNT(NDNUM)=0.0
           II(NDNUM)=I
           JJ(NDNUM)=J
      GO TO 10
200        VMNT(NDNUM)=1.0
           HMNT(NDNUM)=0.0
           II(NDNUM)=I
           JJ(NDNUM)=J
      GO TO 10
300        VMNT(NDNUM)=0.0
           HMNT(NDNUM)=1.0
           II(NDNUM)=I
           JJ(NDNUM)=J
10    CONTINUE
           K=K+1
11    CONTINUE
      RETURN
      END

        SUBROUTINE INPTL

        INCLUDE 'COMMON.FOR'
```

```
C       THIS ROUTINE INITIALIZES ALL NODE LOADS TO ZERO

        DO 5 J=1,NODTOT
        PTLOAD(J)=0.0
   5    CONTINUE
        RETURN
        END

C
C       THE FOLLOWING SUBROUTINE MASSAGES THE INFORMATION CALCUALTED AND
C       STORED BY THE MAIN PROGRAM UNITS SO THAT IT CAN BE PASSED TO THE
C       LINEAR PROGRAMMING SUBROUTINE ZX3LP
C
C
        SUBROUTINE FIXER (A,B,C,PSOL,DSOL,RW,IW)

        DIMENSION A(IA,LMAX),B(IA),C(LMAX),PSOL(MN),DSOL(IA),
     +  RW(IRW1),IW(IW1)


        INCLUDE 'COMMON.FOR'
C
C       PART I--WRITE THE NODAL MOMENTS INEQUALITY CONSTRAINT EQNS
C
        DO 10 I=1,LMAX-1
                DO 20 J=1,LMAX
                        IF(I.EQ.J) THEN
                                A(I,J)=1.
                        ELSE
                                A(I,J)=0.
                        ENDIF
20      CONTINUE
                IF(I.LE.LP2) THEN
                                B(I)=HORZMO*2.
                ELSE
                        B(I)=VERTMO*2.
                ENDIF
10      CONTINUE




C
C       NEXT WRITE THE ADDITIONAL CONSTRAINT EQUATIONS AS SELECTED BY THE
C       USER IN THE LOAD ENTERING SEQUENCE
        IF(IEQNM.EQ.0) GO TO 100
        K=LMAX+IEQNM-1
        DO 30 J=1,LMAX
                LL=1
            IF(J.EQ.LMAX) THEN
               L=2*NODTOT+1
```

```
          ELSE IF(J.LE.LP2) THEN
             L=LIN(J)
          ELSE
             L=LIN(J)+NODTOT
          ENDIF
      DO 40 I=LMAX,K
              A(I,J)=A2(LL,L)
              LL=LL+1
40    CONTINUE
30    CONTINUE


      LL=1
      DO 50 I=LMAX,K
              B(I)=B2(LL)
              LL=LL+1
50    CONTINUE
100   CONTINUE
C
C     NOW TO WRITE THE NODAL EQUATIONS TO MATRIX A
C
      K=LMAX+IEQNM
      DO 60 J=1,LMAX
              LL=1
      IF(J.EQ.LMAX) THEN
          L=2*NODTOT+1
      ELSE IF(J.LE.LP2) THEN
          L=LIN(J)
      ELSE
          L=LIN(J)+NODTOT
      ENDIF
      DO 70 I=K,IA-2
              A(I,J)=FOG(LL,L)
              LL=LL+1
70    CONTINUE
60    CONTINUE
      LL=1
      DO 80 I=K,IA-2
              B(I)=B1(LL)
              LL=LL+1
80    CONTINUE

C
C     FINALLY WE WRITE THE C MATRIX
C
      DO 90 I=I,LMAX-1
              C(I)=0.
90    CONTINUE
              C(LMAX)=1.
C
C     NOW WE ARE READY TO CALL THE LP SUBROUTINE
      CALL ZX3LP(A,IA,B,C,LMAX,M1,M2,S,PSOL,DSOL.RW.IW.IER)
```

```
C
C       NOW TO MASSAGE OUTPUT OF ZX3LP SO THAT IT CAN BE USED
C       IN THE MAIN PROGRAM OUTPUT ROUTINE
C
        DO 99, I=1,LMAX-1
                K=LIN(I)
        IF(I.LE.LP2) THEN
                HMNT(K)=PSOL(I)-HORZMO
        ELSE
                VMNT(K)=PSOL(I)-VERTMO
        ENDIF
99      CONTINUE
                PP=PSOL(LMAX)
C
C       NOW TO RETURN TO THE MAIN PROGRAM
C
        RETURN
        END

         SUBROUTINE EQNS

        INCLUDE 'COMMON.FOR'

C     SUBROUTINE TO CALCULATE AND STORE THE NODAL EQUATIONS OF THE GRID
C
C     FIRST STEP.--INITIALIZE THE ARRAYS
      NODTOT=NODESH*NODESV
      DO 18 I=1,NODEEQ
                        B1(I)=0.
           DO 17 J=1,NODTOT*2+1
               FOG(I,J)=0.
17    CONTINUE
18    CONTINUE
C
C     NOW TO CALCULATE THE NODAL EQUATIONS AND WRITE THEM TO THE ROWS OF
C
           IF(HLEN.GE.VLEN) THEN
                XMULT=HLEN
           ELSE
                XMULT=VLEN
           ENDIF
           M=NODESH+2
           N=NODTOT-NODESH-1
           L=1

      DO 19 I=M,N
           IF(JJ(I).EQ.NODESH.OR.JJ(I).EQ.1) GO TO 19
           R=0.
C     CALCULATE THE COEFFICIENT OF HORZ MO. AT NODE I
           HC=2./HLEN
           R=R+2.*HORZMO/HLEN
           FOG(L,I)=HC
```

```
C        CALCULATE THE COEFFICIENT OF HORZ MO. TO LEFT OF NODE I
             IF(INT(HMNT(I-1)).EQ.0) GO TO 20
             HC=-1./HLEN
             R=R-HORZMO/HLEN
             FOG(L,I-1)=HC
20      CONTINUE
C       CALCULATE THE COEFFICIENT OF THE HORZ MO. TO RIGHT OF NODE I
             IF(INT(HMNT(I+1)).EQ.0) GO TO 21
             HC=-1./HLEN
             R=R-HORZMO/HLEN
             FOG(L,I+1)=HC
21      CONTINUE
C       CALCULATE THE COEFFICIENT OF THE VERT MO. AT NODE I
             VC=2./VLEN
             R=R+2*VERTMO/VLEN
             K1=NODTOT+I
             FOG(L,K1)=VC
C       CALCULATE THE COEFFICIENT OF THE VERT MO. ABOVE NODE I
             K2=I-NODESH
             IF(INT(VMNT(K2)).EQ.0) GO TO 22
             VC=-1./VLEN
             R=R-VERTMO/VLEN
             K3=NODTOT+K2
             FOG(L,K3)=VC
22      CONTINUE
C       CALCULATE THE COEFFICIENT OF THE VERT MO. BELOW NODE I
             K2=I+NODESH
             IF(INT(VMNT(K2)).EQ.0) GO TO 23
             VC=-1./VLEN
             R=R-VERTMO/VLEN
             K3=NODTOT+K2
             FOG(L,K3)=VC
23      CONTINUE
             K2=2*NODTOT+1
             FOG(L,K2)=-PTLOAD(I)
             B1(L)=R
C       CHECK TO SEE IF RHS OF EQUATION IS NEGATIVE, BRANCH ACCORDINGLY
C
        IF(R.LT.0.) GO TO 24
        DO 25 J=1,2*NODTOT+1
             FOG(L,J)=FOG(L,J)*XMULT
25      CONTINUE
             B1(L)=B1(L)*XMULT
        GO TO 26
24      DO 27 J=1,NODTOT*2+1
             FOG(L,J)=FOG(L,J)*-XMULT
27      CONTINUE
             B1(L)=B1(L)*-XMULT
26      CONTINUE
        L=L+1
19      CONTINUE
        RETURN
```

```
      END

        SUBROUTINE CONST

        INCLUDE 'COMMON.FOR'

C
C       THIS SUBROUTINE WRITES THE ADDITIONAL INEQUALITY CONSTRAINT
C       EQUATIONS SELECTED BY THE USER TO ENSURE THAT THE BENDING
C       MOMENT IN A HORIZONTAL OR VERTICAL BEAM DOES NOT EXCEED
C       THE BEAM PLASTIC BENDING MOMENT (IE MOMENT BETWEEN NODES)
C       THIS ALLOWS THE FORMATION OF PLASTIC HINGES AT LOCATIONS OTHER
C       THAN THE NODES
C       THIS ROUTINE ALSO HAS THE CAPABILITY TO CONSTRAIN MOMENTS OF A
C        MEMBER AS HAVING ALREADY REACHED THE YIELD MOMENT.  THAT IS
C       SIMILAR TO HAVING A MEMBER THAT HAS ALREADY FAILED OR HAS MINIMAL
C       STRENGTH LEFT DUE TO EXTREME LOCAL LOADING.  THIS IS
C       ACCOMPLISHED BY MAKING THE ADDITIONAL CONSTRAINT EQUATION
C       AN EQUALITY CONSTRAINT RATHER THAN AN INEQUALITY CONSTRAINT

C       RE-INITIATE HMNT & VMNT ARRAYS FOR USE AS MULTIPLIERS TO
NEQST=0 DETERMINE EXISTANCE OF PARTICULAR MOMENTS BASED ON BOUNDARIES
5       CONTINUE
10      IF(IAUTO.EQ.2) THEN
           PRINT*,'INPUT THE REFERENCE NODE FOR THE ADDITIONAL'
           PRINT*,'CONSTRAINT, AND A 1 FOR HORIZONTAL MEMBER'
           PRINT*,'OR A 2 FOR VERTICAL MEMBER (INTEGERS)'
           PRINT*,'ENTER ZEROS FOR NO CONSTRAINTS'
           READ (5,*) NN,MANDIR
        IF(NN.EQ.0) GO TO 690
           PRINT*,'INPUT THE DISTANCE FROM THE REFERENCE NODE'
           PRINT*,'TO LOCATE THE CONSTRAINT(REAL)'
           READ (5,*) XX
                BHKLOC (NN)=XX
                BVKLOC(NN)=XX
           PRINT*,'ENTER A 1 TO INDICATE THAT THE MEMBER HAS A'
           PRINT*,'FULLY DEVELOPED YIELD MOMENT ALREADY DEVELOPED'
           PRINT*,'NOTE: THIS TYPE MUST BE ENTERED LAST'
           PRINT*,'ENTER A 0 IF THE HINGE IS NOT ALREADY FORMED'
           PRINT*,'(INTEGERS)'
           READ(5,*) NQ
           NEQST=NEQST+NQ
           GO TO 20
        ENDIF

        DO 700 NN=1, NODTOT

20              BHPVAL=0.0
                BHPLOC=0.0
                BVPVAL=0.0
                BVPLOC=0.0
```

```
            BHLAV=0.0
            BHLBV=0.0
            BVLAV=0.0
            BVLBV=0.0

      IF(IAUTO.EQ.2) THEN
            BHKMNT(NN)=HORZMO
            BVKMNT(NN)=VERTMO
      ENDIF
            DO 100 J=1,LPT
              IF (LD1ND(J).EQ.NN.AND. D1LOC(J) .EQ.0.0)
     +               GO TO 100
              IF (LD1ND(J).EQ.NN.AND. D1LOC(J) .GT. 0.0
     +            .AND. BHKMNT(NN).GE.HORZMO) THEN

                  BHPVAL=D1VAL(J)
                  BHPLOC=D1LOC(J)

              ELSE IF (LD1ND(J).EQ.NN.AND.D1LOC(J).LT.0.0
     +                 .AND.BVKMNT(NN).GE.VERTMO) THEN

                  BVPVAL=ABS(D1VAL(J))
                  BVPLOC=ABS(D1LOC(J))

              ENDIF

100         CONTINUE


            DO 200 J=1,LNLDN

              IF(LD2ND(J).EQ.NN .AND. ((D2BVL(J) .GT. 0.0.OR.
     +            D2AVL(J).GT.0.0)) .AND. BHKMNT(NN).GE.HORZMO)
     +               THEN

                  BHLAV=D2AVL(J)
                  BHLBV=D2BVL(J)

              ELSE IF (LD2ND(J).EQ.NN .AND. (D2BVL(J) .LT. 0.
     +                 .OR.D2AVL(J).LT.0.0).AND.BVKMNT(NN)
     +                 .GE.VERTMO) THEN

                  BVLAV=ABS(D2AVL(J))
                  BVLBV=ABS(D2BVL(J))

              ENDIF

200         CONTINUE

C  IF THERE ARE NO LOADS ASSOCIATED WITH THIS NODE, CONTNUE

      IF(BHPVAL.EQ.0.0 .AND. BVPVAL .EQ. 0.0 .AND. BHLAV .EQ.
```

```
     +    0.0 .AND. BHLBV .EQ. 0.0 .AND. BVLAV .EQ. 0.0 .AND.
     +    BVLBV .EQ. 0.0)  GO TO 680

      IF(BHPVAL.EQ.0.AND.BHLAV.EQ.0.0.AND.BHLBV.EQ.0.0)
     +     GO TO 450
C  CHECK FOR VERTICAL OR HORIZONTAL FOR MANUAL MODE
      IF(MANDIR.EQ.2) GO TO 450
C  HORIZONTAL BEAM CALCULATIONS
      XL=HLEN
      A=BHPLOC
      Q=BHPVAL
      STEP=HLEN/1000.0
      XX=BHKLOC(NN)



C   INCREASING OR CONSTANT LINE LOAD WITH MAX MOMENT TO THE LEFT OF
C    OR AT A POINT LOAD
      IF(BHLAV.LE.BHLBV.AND.XX.LE.A) THEN
              P=BHLBV-BHLAV
              W=BHLAV


      X1=(1.-XX/XL)*HMNT(NN)
      X2=(XX/XL)*HMNT(NN+1)
      X3=(1.-A/XL)*Q*XX+.5*W*XL*XX+P*XL*XX/6.0
     +     -.5*W*XX**2-P*XX**3/(6.0*XL)
      X4=HORZMO*(1.+(1.-XX/XL)*HMNT(NN)+(XX/XL)
     +             *HMNT(NN+1))
      IDIR=1
              XDCNST(NN)=XX
              IF(NQ.EQ.1)THEN
                 NDCNST(NN)=-NN
              ELSE
                 NDCNST(NN)=NN
              ENDIF

      CALL CONST2 (X1,X2,X3,X4,IDIR,NN)

C   INCREASING OR CONSTANT LINE LOAD WITH MAX MOMENT
C   TO RIGHT OF POINT LOAD OR NO POINT LOAD
      ELSE IF(BHLAV.LE.BHLBV.AND.XX.GT.A) THEN

              P=BHLBV-BHLAV
              W=BHLAV
              XXX=XL-XX


      X1=(XXX/XL)*HMNT(NN)
      X2=(1.-XXX/XL)*HMNT(NN+1)
      X3=(A/XL)*Q*XXX+.5*W*XL*XXX+P*XL*XXX/3.
     +-.5*W*XXX**2 -.5*P*XXX**2+P*XXX**3/(6.0*XL)
```

```
      X4=HORZMO*(1.+(1.-XXX/XL)*HMNT(NN+1)+(XXX/XL)
     +            *HMNT(NN))
      IDIR=1
              XD        IF(NQ.EQ.1)THEN
                 NDCNST(NN)=-NN
              ELSE
                 NDCNST(NN)=NN
              ENDIF

      CALL CONST2 (X1,X2,X3,X4,IDIR,NN)


C    DECREASING LINE LOAD WITH MAX MOMENT TO THE LEFT OF OR AT
C     A POINT LOAD OR WITH NO POINT LOAD

      ELSE IF (BHLBV.LT.BHLAV.AND.XX.LE.A) THEN
              P=BHLAV-BHLBV
              W=BHLBV

      X1=(1.-XX/XL)*HMNT(NN)
      X2=(XX/XL)*HMNT(NN+1)
      X3=(1.-A/XL)*Q*XX+.5*W*XL*XX+P*XL*XX/3.-.5*W*XX**2
     +      +P*XX**3/(6.0*XL)
      X4=HORZMO*(1.+(1.-XX/XL)*HMNT(NN)+(XX/XL)*HMNT(NN+1))
      IDIR=1
              XDCNST(NN)=XX
              IF(NQ.EQ.1)THEN
                 NDCNST(NN)=-NN
              ELSE
                 NDCNST(NN)=NN
              ENDIF

      CALL CONST2 (X1,X2,X3,X4,IDIR,NN)

C    DECREASING LINE LOAD WITH MAX MOMENT TO THE RIGHT OF THE
C     POINT LOAD

      ELSE IF(BHLBV.LT.BHLAV.AND.XX.GT.A) THEN

              P=BHLAV-BHVBV
              W=BHLBV
              XXX=XL-X

      X1=(XXX/XL)*HMNT(NN)
      X2=(1.-XXX/XL)*HMNT(NN+1)
      X3=(A/XL)*Q*XXX+.5*W*XL*XXX+P*XL*XXX/6.-.5*W*XXX**2
     +      +P*XXX**3/(6.0*XL)
      X4=HORZMO*(1.+(1.-XXX/XL)*HMNT(NN+1)+(XXX/XL)*HMNT(NN))
      IDIR=1
              XDCNST(NN)=XX
              IF(NQ.EQ.1)THEN
                 NDCNST(NN)=-NN
```

```
               ELSE
                  NDCNST(NN)=NN
               ENDIF

      CALL CONST2 (X1,X2,X3,X4,IDIR,NN)
      ENDIF

C  VERTICAL MEMBER CALCULATIONS

450   CONTINUE

C     SKIP THIS SECTION IF THERE ARE NO LOADS ON THE VERT MEMBER
      IF((BVPLOC.EQ.0.0.OR.BVPVAL.EQ.0.0).AND.BVLAV.EQ.0.0
     +    .AND.BVLBV.EQ.0.0) GO TO 680
C     SKIP THIS SECTION IN MANUAL IF HORIZONTAL CONSTRAINT
      IF(MANDIR.EQ.1) GO TO 680
      XL=VLEN
      A=BVPLOC
      Q=BVPVAL
      XX=BVKLOC(NN)



C   INCREASING LINE LOAD WITH MAX MOMENT TO THE LEFT OF
C    OR AT A POINT LOAD
      IF(BVLAV.LE.BVLBV.AND.XX.LE.A) THEN
              P=BVLBV-BVLAV
              W=BVLAV

      X1=(1.-XX/XL)*VMNT(NN)
      X2=(XX/XL)*VMNT(NN+NODESH)
      X3=(1.-A/XL)*Q*XX+.5*W*XL*XX+P*XL*XX/6.-.5*W*XX**2
     +    -P*XX**3/(6.0*XL)
      X4=VERTMO*(1.+(1.-XX/XL)*VMNT(NN)+(XX/XL)*VMNT(NN+NODESH))
      IDIR=2
              XDCNST(NN)=-XX
              IF(NQ.EQ.1)THEN
                 NDCNST(NN)=-NN
              ELSE
                 NDCNST(NN)=NN
              ENDIF

      CALL CONST2 (X1,X2,X3,X4,IDIR,NN)

C   INCREASING LINE LOAD WITH MAX MOMENT TO RIGHT OF
C   POINT LOAD OR NO PT LOAD
      ELSE IF(BVLAV.LE.BVLBV.AND.XX.GT.A) THEN

              P=BVLBV-BVLAV
              W=BVLAV
              XXX=XL-XX
```

```
      X1=(XXX/XL)*VMNT(NN)
      X2=(1.-XXX/XL)*VMNT(NN+NODESH)
      X3=(A/XL)*Q*XXX+.5*W*XL*XXX+P*XL*XXX/3.-.5*W*XXX**2
     +      -.5*P*XXX**2+P*XXX**3/(6.0*XL)
      X4=VERTMO*(1.+(1.-XXX/XL)*VMNT(NN+NODESH)
     +                      +(XXX/XL)*VMNT(NN))
      IDIR=2
              XDCNST(NN)=-XX
              IF(NQ.EQ.1)THEN
                 NDCNST(NN)=-NN
              ELSE
                 NDCNST(NN)=NN
              ENDIF

      CALL CONST2 (X1,X2,X3,X4,IDIR,NN)


C   DECREASING LINE LOAD WITH MAX MOMENT TO THE LEFT OF OR AT
C    A POINT LOAD OR WITH NO POINT LOAD

      ELSE IF (BVLBV.LT.BVLAV.AND.XX.LE.A) THEN
              P=BVLAV-BVLBV
              W=BVLBV

      XL)*VMNT(NN+NODESH)
      X3=(1.-A/XL)*Q*XX+.5*W*XL*XX+P*XL*XX/3.-.5*W*XX**2
     +      +P*XX**3/(6.0*XL)
      X4=VERTMO*(1.+(1.-XX/XL)*VMNT(NN)+(XX/XL)*VMNT(NN+NODESH))
      IDIR=2
              XDCNST(NN)=-XX
              IF(NQ.EQ.1)THEN
                 NDCNST(NN)=-NN
              ELSE
                 NDCNST(NN)=NN
              ENDIF

      CALL CONST2 (X1,X2,X3,X4,IDIR,NN)

C   DECREASING LINE LOAD WITH MAX MOMENT TO THE RIGHT OF THE
C    POINT LOAD

      ELSE IF(BVLBV.LT.BVLAV.AND.XX.GT.A) THEN

              P=BVLAV-BVVBV
              W=BVLBV
              XXX=XL-X

      X1=(XXX/XL)*VMNT(NN)
      X2=(1.-XXX/XL)*VMNT(NN+NODESH)
      X3=(A/XL)*Q*XXX+.5*W*XL*XXX+P*XL*XXX/6.-.5*W*XXX**2
     +      +P*XXX**3/(6.0*XL)
      X4=VERTMO*(1.+(1.-XXX/XL)*VMNT(NN+NODESH)
```

```
     +                    +(XXX/XL)*VMNT(NN))
        IDIR=2
                XDCNST(NN)=-XX
                IF(NQ.EQ.1)THEN
                   NDCNST(NN)=-NN
                ELSE
                   NDCNST(NN)=NN
                ENDIF

        CALL CONST2 (X1,X2,X3,X4,IDIR,NN)

        ENDIF

680     IF(BHPVAL.EQ.0.0.AND.BVPVAL.EQ.0.0.AND.
     +  BHLAV.EQ.0.0.AND.BHLBV.EQ.0.0.AND.
     +  BVLAV.EQ.0.0.AND.BVLBV.EQ.0.0.AND.
     +  IAUTO.EQ.2) THEN
           IF(MANDIR.EQ.1)THEN
                XMULT1=HMNT(NN)
                XMULT2=HMNT(NN+1)
                XL=HLEN
                XMO=HORZMO
                IDIR=1
           ELSE IF(MANDIR.EQ.2)THEN
                XMULT1=VMNT(NN)
                XMULT2=HMNT(NN+NODESH)
                XL=VLEN
                XMO=VERTMO
                IDIR=2
           ENDIF
          X1=(1.-XX/XL)*XMULT1
          X2=(XX/XL)*XMULT2
          X3=0.0
          X4=XMO*(1.+(1.-XX/XL)*XMULT1+XX/XL*XMULT2)
                IF(IDIR.EQ.1)THEN
                    XDCNST(NN)=XX
                ELSE IF(IDIR.EQ.2) THEN
                    XDCNST(NN)=-XX
                ENDIF

                IF(NQ.EQ.1) THEN
                    NDCNST(NN)=-NN
                ELSE
                    NDCNST(NN)=NN
                ENDIF
        CALL CONST2(X1,X2,X3,X4,IDIR,NN)
        ENDIF


690     IF(IAUTO.EQ.2) THEN
           PRINT*,'     '
           PRINT*,'ENTER 1 FOR ADDITIONAL CONSTRAINTS OR 0'
```

```
            PRINT*,'FOR NO ADDITIONAL RESTRAINTS REQUIRED'
            READ (5,*) IIII
            IF(IIII.EQ.1) GO TO 10
            GOTO 720
         ENDIF

700      CONTINUE

720      RETURN
         END
         SUBROUTINE CONST2 (X1,X2,X3,X4,IDIR,NDREF)

         INCLUDE 'COMMON.FOR'
         L=2*NODTOT
500      IEQNM=IEQNM+1
         N2=NDREF+1

         IF(IDIR.EQ.2) GO TO 20
         DO 10 I=1,NDEF-1
                 A2(IEQNM,I)=0.
10       CONTINUE
                 A2(IEQNM,NDREF)=X1
                 A2(IEQNM,N2)=X2
         DO 15 I=N2+1,L
                 A2(IEQNM,I)=0.
15       CONTINUE
                 A2(IEQNM,L+1)=X3
                 B2(IEQNM)=X4
         GO TO 40

20       CONTINUE
         J=NODTOT+NDREF
         K=J+NODESH
         DO 25 I=1,J-1
                 A2(IEQNM,I)=0.
25       CONTINUE
                 A2(IEQNM,J)=X1
         DO 30 I=J+1,K-1
                 A2(IEQNM,I)=0.
30       CONTINUE
                 A2(IEQNM,K)=X2
         DO 35 I=K+1,L
                 A2(IEQNM,I)=0.
35       CONTINUE
                 A2(IEQNM,L+1)=X3
                 B2(IEQNM)=X4
40       CONTINUE

         RETURN
         END
```

```
      SUBROUTINE BEAMCK

      INCLUDE 'COMMON.FOR'
      CHARACTER* 9 NAME,NAME1,NAME2,NAME3,NAME4,NAME5
      CHARACTER* 9 NAME6,NAME7,NAME8

      NAME1='I/NODE#'
      NAME='LD2ND #'
      NAME2='D2AVL'
AME5='BHLBV'NAME3='D2B
      NAME6='BVLAV'
      NAME7='BVLBV'
      NAME8='NNN/LNLDN'

      DO 999 I=1,LNLDN
999   CONTINUE


      DO 700 I=1, NODTOT
              BHKMNT(I)=0.0
              BHKLOC(I)=0.0
              BVKMNT(I)=0.0
              BVKLOC(I)=0.0

              BHPVAL=0.0
              BHPLOC=0.0
              BVPVAL=0.0
              BVPLOC=0.0

              BHLAV=0.0
              BHLBV=0.0
              BVLAV=0.0
              BVLBV=0.0


5000  FORMAT(2X,A9,1X,I4,5X,A9,G12.2,3X,A9,G12.2,3X,A9,
     +        G12.2,3X,A9,G12.2)


              DO 100 J=1,LPT
                IF (LD1ND(J).EQ.I.AND. D1LOC(J) .EQ.0.0) GO TO 100
                IF (LD1ND(J).EQ.I.AND. D1LOC(J) .GT. 0.0) THEN

                    BHPVAL=D1VAL(J)*PP
                    BHPLOC=D1LOC(J)

                ELSE IF (LD1ND(J).EQ.I.AND.D1LOC(J).LT.0.0) THEN
```

```
                BVPVAL=ABS(D1VAL(J))*PP
                BVPLOC=ABS(D1LOC(J))

            ENDIF


100         CONTINUE


            DO 200 NNN=1,LNLDN
              IF((LD2ND(NNN).EQ.I) .AND. (D2BVL(NNN) .GT. 0.0.OR.
     +          D2AVL(NNN).GT.0.0)) THEN

                BHLAV=D2AVL(NNN)*PP
                BHLBV=D2BVL(NNN)*PP

              ELSE IF((LD2ND(NNN).EQ.I).AND.(D2BVL(NNN).LT.0.0.OR.
     +          D2AVL(NNN).LT.0.0)) THEN

                BVLAV=ABS(D2AVL(NNN))*PP
                BVLBV=ABS(D2BVL(NNN))*PP

              ENDIF

200         CONTINUE

C  IF THERE ARE NO LOADS ASSOCIATED WITH THIS NODE, CONTNUE

      IF(BHPVAL.EQ.0.0 .AND. BVPVAL .EQ. 0.0 .AND. BHLAV .EQ.
     +  0.0 .AND. BHLBV .EQ. 0.0 .AND. BVLAV .EQ. 0.0 .AND.
     +  BVLBV .EQ. 0.0)  GO TO 450


      IF(BHPVAL.EQ.0.AND.BHLAV.EQ.0.0.AND.BHLBV.EQ.0.0)
     +    GO TO 450

C  HORIZONTAL BEAM CALCULATIONS
      XL=HLEN
      A=BHPLOC
      Q=BHPVAL
      BM1=HMNT(I)
      BM2=HMNT(I+1)
      STEP=HLEN/1000.0

      BMXA=0.0
      BMXB=0.0
      BMX=0.0

      DO 300 NNN=1,1000
      X=NNN*STEP
```

```
      XX=X
C   SET X FOR SECOND INTERVAL IF TO RIGHT OF POINT LOAD
      IF(A.NE.0.0.AND.Q.NE.0.0.AND.X.GT.A) THEN
         X=X-A
      END IF

C   INCREASING OR CONSTANT LINE LOAD
      IF(BHLAV.LE.BHLBV) THEN
              P=BHLBV-BHLAV
              W=BHLAV
              RCOM=(BM2-BM1)*X/XL+W*XL*X*.5+
     +             P*XL*X/6.0-.5*W*X**2-P*X**3/
     +             (6.0*XL)
C   DECREASING LINE LOAD
      ELSE
              P=BHLAV-BHLBV
              W=BHLBV
              RCOM=(BM2-BM1)*X/XL+W*XL*X*.5+
     +             P*XL*X/3.0-.5*W*X**2-P*X**2*.5
     +             +P*X**3/(6.0*XL)
      ENDIF

      IF(Q.EQ.0.0.OR.A.EQ.0.0) THEN
              BMX=RCOM+BM1
      ELSE IF (X.LE.A) THEN
              BMXA=RCOM+BM1+(1-A/XL)*Q*X
      ELSE
              BMXB=BMXA+RCOM-A*Q*X/XL
      ENDIF
           IF(ABS(BMX).LT.ABS(BMXA).OR.ABS(BMX)
     +        .LT.ABS(BMXB)) THEN
              IF(ABS(BMXA).GT.ABS(BMXB)) THEN
                 BMX=BMXA
              ELSE
                 BMX=BMXB
              ENDIF
           ENDIF

           IF(ABS(BHKMNT(I)).LT.ABS(BMX)) THEN
              BHKMNT(I)=BMX
              BHKLOC(I)=XX
           ENDIF

300        CONTINUE


450     CONTINUE

C    SET MAX MOMENT TO CONSTRAINED VALUE IF MEMBER HAS EXTRA
C    CONSTRAINT EQUATION FOR IT AND CONSTRAINT IS LIMITING

      IF(XDCNST(I).GT.0.0.AND.(BHKMNT(I).GE.HORZMO
```

```
     +     .OR.NDCNST(I).LT.0))THEN
             BHKMNT(I)=HORZMO
             BHKLOC(I)=XDCNST(I)
           ENDIF

C          VERTICAL MEMBER CALCULATIONS
C          SKIP THIS SECTION IF THERE ARE NO LOADS ON THE VERT MEMBER

           IF((BVPLOC.EQ.0.OR.BVPVAL.EQ.0.0).AND.(BVLAV.EQ.0.0
     +             .AND.BVLBV.EQ.0.0)) GO TO 610


           A=BVPLOC
           Q=BVPVAL
           BM1=VMNT(I)
           BM2=VMNT(I+NODESH)
           STEP=VLEN/1000.0
           XL=VLEN

           BMXA=0.0
           BMXB=0.0
           BMX=0.0


           DO 600 NNN=1,1000
           X=NNN*STEP
           XX=X
C      SET X FOR SECOND INTERVAL IF TO RIGHT OF POINT LOAD
           IF(A.NE.0.0.AND.Q.NE.0.0.AND.X.GT.A) THEN


C      INCREASING OR CONSTANT LINE LOAD
           IF(BVLAV.LE.BVLBV) THEN
                   P=BVLBV-BVLAV
                   W=BVLAV


                   RCOM=(BM2-BM1)*(X/XL)+W*XL*X*.5+
     +                     (P*XL*X/6.0)-(.5*W*X**2)-
     +                     (P*X**3/(6.0*XL))
C      DECREASING LINE LOAD
           ELSE
                   P=BVLAV-BVLBV
                   W=BVLBV
                   RCOM=(BM2-BM1)*X/XL+W*XL*X*.5+
     +                     P*XL*X/3.0-.5*W*X**2-P*X**2*.5
     +                     +P*X**3/(6.0*XL)
           ENDIF

           IF(Q.EQ.0.0.OR.A.EQ.0.0) THEN
                   BMX=RCOM+BM1
           ELSE IF (X.LE.A) THEN
```

```
              BMXA=RCOM+BM1+(1-A/XL)*Q*X
      ELSE
              BMXB=BMXA+RCOM-A*Q*X/XL
      ENDIF
           IF(ABS(BMX).LT.ABS(BMXA).OR.ABS(BMX)
     +       .LT.ABS(BMXB)) THEN
              IF(ABS(BMXA).GT.ABS(BMXB)) THEN
                 BMX=BMXA
              ELSE
                 BMX=BMXB
              ENDIF
           ENDIF

           IF(ABS(BVKMNT(I)).LT.ABS(BMX)) THEN
              BVKMNT(I)=BMX
              BVKLOC(I)=X
           ENDIF

600      CONTINUE
610      CONTINUE

      IF(XDCNST(I).LT.0.0 .AND. (BVKMNT(I) .GE. VERTMO
     +     .OR. NDCNST(I).LT.0.0)) THEN
              BVKMNT(I)=VERTMO
              BVKLOC(I)=ABS(XDCNST(I))
      ENDIF

700   CONTINUE

      RETURN
      END



      SUBROUTINE OUTPUT

      INCLUDE 'COMMON.FOR'

      CHARACTER*15 SIDE,BBCC,DIR,FLAG
      CHARACTER*20 CNST

C     THIS ROUTINE OUTPUTS THE GRID LOADING SCHEME AND ALL MOMENTS AND LO
C     FACTORS

C     CHECK ITERATION NUMBER IF IT IS '1' THEN PRINT GRID DATA
      IF(ITERAT.NE.1) GO TO 50
C     PRINT OUT GRID SIZE
      WRITE (15,1000)
      WRITE (15,1100) NODESH,HLEN
      WRITE (15,1200) NODESV,VLEN

C     PRINT OUT THE GRID BOUNDARY CONDITIONS
```

```
      WRITE (15,1300)
      DO HEN
                 IBOUND=ITOP
                 SIDE='TOP'
            ELSE IF (I.EQ.2) THEN
                 IBOUND=IBTM
                 SIDE='BOTTOM'
            ELSE IF (I.EQ.3) THEN
                 IBOUND=ILFT
                 SIDE='LEFT'
            ELSE
                 IBOUND=IRGT
                 SIDE='RIGHT'
            ENDIF

            IF (IBOUND .EQ. 1) THEN
                BBCC='SIMPLE SUPORTED'
            ELSE
                BBCC='CLAMPED'
            ENDIF

         WRITE (15,1400) SIDE,BBCC

15    CONTINUE
        PRINT*,HORZMO,VERTMO
C   WRITE THE ZX3LP ERROR NUMBER
        WRITE(15,1425) HORZMO,VERTMO

1425    FORMAT(////,' HORZ BEAM PLASTIC BENDING MOMENT = ',G12.2,
     +//,' VERT BEAM PLASTIC BENDING MOMENT = ',G12.2)


         WRITE(15,1450)IER

1450    FORMAT(//,5X,'ZX3LP ERROR NUMBER = ',I4)
C     PRINT OUT ALL THE POINT LOADS ON THE GRID
      WRITE (15,1500)
      WRITE (15,1600)

      DO 25 I=1,LPT
        IF(ABS(D1LOC(I)).LT.0.01) THEN
           DIR='ON THE NODE'
        ELSE IF (D1LOC    DIR='TO THE RIGHT'
        ELSE
           DIR='BELOW'
        ENDIF
      WRITE (15,1700) LD1ND(I),DIR,ABS(D1LOC(I)),D1VAL(I),I
25    CONTINUE


C     PRINT OUT LINE LOADS
      WRITE (15,1800)
```

```
      DO 30 I=1,LNLDN
      IF (D2BVL(I) .GT. 0.0) THEN
           NB=LD2ND(I)+1
      ELSE
           NB=LD2ND(I)+NODESH
      ENDIF
      WRITE (15,1900) LD2ND(I),NB,ABS(D2AVL(I)),ABS(D2BVL(I)),I
30    CONTINUE


1000  FORMAT ('1',5X,'GRID CHARACTERISTICS AND LOADING',///)
1100  FORMAT (6X,'NUMBER OF NODES HORIZONTAL = ',I3,5X,
     +'HORIZONTAL LENGTH =',F6.2)
1200  FORMAT (6X,'NUMBER OF NODES VERTICAL = ',I3,5X,
     +'VERTICAL LENGTH =',F6.2)
1300  FORMAT ('0',5X,'BOUNDARY CONDITIONS',/)
1400  FORMAT (6X,A6,T25,A15)
1500  FORMAT (///,6X,'POINT LOADS ADDED TO THE GRID',/)
1600  FORMAT (3X,'REF NODE',T14,'DIR FM NODE',T35,
     +'DISTANCE FROM NODE',T57,'MAGNITUDE',T70,
     +'POINT LOAD NUMBER')
1700    FORMAT (6X,I3,T14,A13,T45,F6.2,T58,F10.2,T72,I3)
1800    FORMAT   (8(/),5X,'LINE  LOADS',//,6X,'BETWEEN  NODES',
     +T30,'REF NODE MAG',T45,'OTHER NODE MAG',
     +T65,'LINE LOAD NUMBER',/)
1900  FORMAT (6X,I3,' AND ',I3,T28,F10.2,T44,F10.2,T70,I3)



40    CONTINUE

C     PRINT OUT THE FINAL NODAL LOADING THAT CALCULATIONS ARE BASED ON

      WRITE (15,2300)
2300  FORMAT (8(/),5X,'NODE LOADS USED IN CALCULATIONS',//)
      WRITE (15,2400)
2400  FORMAT (6X,'NODE NUMBER',T20,'NODE LOAD')
           DO 50 I=1,NODTOT

           WRITE(15,2500) I,PTLOAD(I)
2500  FORMAT (10X,I3,T21,F8.2)

50    CONTINUE
        WRITE(15,2550)ITERAT

C     PRINT OUT THE EXTRA CONSTRAINT DATA
        WRITE (15,2525)
2525    FORMAT(//,2X,'ADDITIONAL CONSTRAINTS USED',/)

        IF(IAUTO.EQ.1) THEN
                WRITE(15,2527)
        ELSE IF(IAUTO.EQ.2)THEN
```

```
                WRITE(15,2528)
        ELSE
                WRITE(15,2529)
        ENDIF
2527    FORMAT(2X,'AUTOMATIC CONSTRAINTS WERE USED')
2528    FORMAT(2X,'MANUAL CONSTRAINTS WERE USED')
2529    FORMAT(2X,'NO ADDITIONAL CONSTRAINTS WERE USED')

        WRITE(15,2540)
2540    FORMAT(6X,'BETWEEN NODES',3X,'LOCATION',
     +                '    TYPE USED')

        DO 57, I=1,NODTOT
           NOD1=ABS(NDCNST(I))
        IF(XDCNST(I).GT.0.0) THEN
           NOD2=ABS(NDCNST(I))+1
        ELSE IF(XDCNST(I).LT.0.0) THEN
           NOD2=ABS(NDCNST(I))+NODESH
                ELSE
           GO TO 57
        ENDIF
        IF(NDCNST(I).GT.0.0) THEN
           CNST=' INEQUALITY '
        ELSE
           CNST=' EQUALITY '
        ENDIF

        WRITE(15,2545)NOD1,NOD2,ABS(XDCNST(I)),CNST

57      CONTINUE

2545    FORMAT(6X,I3,' AND ',I3,6X,G10.2,A20)

C     PRINT OUT MOMENTS AT EACH NODE AND THE LOAD FACTOR
2550    FORMAT(2X,//,'***** ITERATION NUMBER IS',I4,' *****',//)
      WRITE (15,2600)
2600  FORMAT (2(/),6X,'MOMENTS AND  LOAD FACTOR')
      WRITE (15,2700)
2700  FORMAT (//,6X,'NODE NUMBER',T25,'HORIZ MNT',T40,'VERT MNT')

      DO 60 I=1,NODTOT
      WRITE(15,2800)I,HMNT(I),VMNT(I)
2800  FORMAT (8X,I3,T26,E8.2,T42,E8.2)

60    CONTINUE

      WRITE (15,2900) PP
2900  FORMAT (///,6X,'THE GRID LOAD FACTOR = ',G8.2)


C    PRINT OUT MAX MOMENT AND LOCATION FOR EACH MEMBER
```

```
      WRITE(15,3000)
3000  FORMAT(3(/),////,6X,'MAXIMUM MOMENTS AND LOCATION
     + ON LOADED MEMBERS',/)
      WRITE(15,4050)
4050  FORMAT(6X,'BETWEEN NODES',T30,'MAGNITUDE',T45,
     +  'LOCATION')

      WRITE(15,4100)
4100  FORMAT(6X,'HORIZONTAL MEMBERS')
      DO 70 I=1,NODTOT
         IF(BHKMNT(I).EQ.0.0) GO TO 70
         IF(ABS(BHKMNT(I)).GT.(HORZMO+.01)) THEN
              FLAG='**'
         ELSE
              FLAG=' '
         ENDIF
         WRITE(15,4200) I,I+1,BHKMNT(I),BHKLOC(I),FLAG
70    CONTINUE


      WRITE(15,4300)
4300  FORMAT(6X,'VERTICAL MEMBERS')
      DO 80 I=1,NODTOT
        IF(BVKMNT(I).EQ.0.0) GO TO 80
        IF(ABS(BVKMNT(I)).GT.(VERTMO+.01)) THEN
              FLAG ='**'
        ELSE
              FLAG ='  '
        ENDIF
        WRITE(15,4200)I,I+NODESH,BVKMNT(I),BVKLOC(I),FLAG
80    CONTINUE
      WRITE (15,5000)
4200  FORMAT(6X,I3,' AND ',I3,T28,G10.4,8X,G10.4,2X,A15)
5000  FORMAT(////,5X,'** INDICATES > YIELD MOMENT')
    RETURN
    END
```

APPENDIX B

IMSL LINEAR PROGRAMMING ROUTINE

The linear programming routine used by GRIDS is taken from the International Mathematics and Statistics Library (IMSL). GRIDS uses routine ZX3LP. The attached pages describe the routine input requirements and output formatting and variables. This information is provided for a better understanding of the dimensioning process that GRIDS performs. These pages have been reproduced from the IMSL library reference manual.

PURPOSE - Solve the linear programming problem via the revise simplex algorithm - easy to use version

USAGE - CALL ZX3LP (A,IA,B,C,N,M1,M2,S,PSOL,DSOL,RW,IW,IER)

ARGUMENTS

A - Matrix of dimension M1+M2+2 by N containing the coefficients ot the M1 rows followed by the coefficients of the M2 equality constraints (input). The last two rows are used only as working storage.

IA - Row dimension of matrix A exactly as specified in the dimension statement in the calling program. (input) Two rows of A are required for working storage, and therefore, IA must not be less than M1+M2+2.

B - Vector of length M1+M2+2 containing the right hand sides of the inequality constraints.(input) The last two elements of B are used as working storage.

C - Vector of length N containing the coefficients of the objective function. (input)

N - Number of unknowns in the model. (input)

M1 - Number of inequality constraints. (input)

M2 - Number of equality constraints. (input)

S - Value of the objective function. (output)

PSOL - Vector of length N containing the primal solution. (output) PSOL is also used as work storage and therefore must have length at least MAX(n,M1+M2).

DSOL - Vector of length M1+M2+2 containing the DUAL solution. (output) That is, DSOL(1), ..., DSOL(M1+M2) contain the solution to the problem MIN BT*Y subject to AT*Y is greater than or equal to C and Y greater than or equal to 0 where AT = A-transpose and BT = B-transpose. When the primal problem has equality constraints, the corresponding components of the dual solution are unconstrained. DSOL(M1+M2+1) and DSOL(M1+M2+2) are used as working storage.

RW - Work Vector of length (M1+M2+2)*(M1+M2+2) + 3*M1 + 2*M2+4.

IW - Work Vector of length 2*M2 + 3*M1 + 4.

IER - ERROR indicator. (output)
Terminal Error
IER = 130 Indicates that IA is less than M1+M2+2.
IER = 131 Indicates that the cost criterion has unbounded values.
IER = 132 Indicated that the Maximum number of iterations was reached in ZXOLP.
IER = 133 Indicated that no feasible solution exists.
Warnig (with fix)
IER = 70 Indicates that some artifical variables remain in the solution basis at a zero level after phase 1. This condition can be caused by having redundant constraints. Nevertheless, a solution is computed and returned in PSOL and DSOL.

<u>Algorithm</u>

To solve the linear programming problem,

$$\text{Maximize} \quad C_1 PSOL_1 + \ldots + C_N PSOL_N = SP$$

subject to

$$a_{i1} PSOL_1 + \ldots + a_{iN} PSOL_N \leq B_i \qquad i=1,\ldots,M_1$$

$$a_{i1} PSOL_1 + \ldots + a_{iN} PSOL_N = B_i \qquad i=M_1+1,\ldots,M$$

$$PSOL_j \geq 0 \qquad j=1,\ldots,N$$

where M = M1 + M2.

The DUAL linear programming problem is,

$$\text{Minimize} \quad b_1 DSOL_1 + \ldots + B_M DSOL_N = SD$$

subject to

$$a_{1j} DSOL_1 + \ldots + a_{Mj} DSOL_M \geq C_j \qquad j=1,\ldots,N$$

$$DSOL_i \geq 0 \qquad i=1,\ldots,M_1$$

$DSOL_i$ unrtestricted in sign when $i = M_1+1,\ldots,M$

ZX3LP computes the solution to the primal problem, PSOL, the solution to the dual problem, DSOL, and the values of the objective function S=SP=SD.

ZX3LP calls ZX0LP which solve the linear programming problem by the revised simplex method.

where M = M1 + M2.

APPENDIX C

SAMPLE PROBLEMS AND RESULTS

EXAMPLE 1

GRID CHARACTERISTICS AND LOADING

NUMBER OF NODES HORIZONTAL = 6   HORIZONTAL LENGTH = 10.00
NUMBER OF NODES VERTICAL = 5   VERTICAL LENGTH = 10.00

BOUNDARY CONDITIONS

TOP	CLAMPED
BOTTOM	CLAMPED
LEFT	CLAMPED
RIGHT	CLAMPED

HORZ BEAM PLASTIC BENDING MOMENT = 0.10E+05

VERT BEAM PLASTIC BENDING MOMENT = 0.10E+05

ZX3LP ERROR NUMBER = 0

POINT LOADS ADDED TO THE GRID

REF NODE	DIR FM NODE	DISTANCE FROM NODE	MAGNITUDE	LOAD NUMBER
8	ON THE NODE	0.00	1.00	1

LINE LOADS

BETWEEN NODES	REF NODE MAG	OTHER NODE MAG	LINE LOAD NUMBER

NODE LOADS USED IN CALCULATIONS

NODE NUMBER	NODE LOAD
1	0.00
2	0.00
3	0.00
4	0.00
5	0.00
6	0.00
7	0.00
8	1.00
9	0.00
10	0.00
11	0.00
12	0.00
13	0.00
14	0.00
15	0.00
16	0.00
17	0.00
18	0.00
19	0.00
20	0.00
21	0.00
22	0.00
23	0.00
24	0.00
25	0.00
26	0.00
27	0.00
28	0.00
29	0.00
30	0.00

**** ITERATION NUMBER IS 1 *****

ADDITIONAL CONSTRAINTS USED

NO ADDITIONAL CONSTRAINTS WERE USED

BETWEEN NODES	LOCATION	TYPE USED

MOMENTS AND LOAD FACTOR

NODE NUMBER	HORIZ MNT	VERT MNT
1	0.00E+00	0.00E+00
2	0.00E+00	-.10E+05
3	0.00E+00	-.10E+05
4	0.00E+00	-.10E+05
5	0.00E+00	-.10E+05
6	0.00E+00	0.00E+00
7	-.10E+05	0.00E+00
8	0.10E+05	0.10E+05
9	-.10E+05	0.30E+04
10	-.40E+04	-.10E+05
11	-.70E+04	-.10E+05
12	-.10E+05	0.00E+00
13	-.10E+05	0.00E+00
14	0.84E+04	-.10E+05
15	0.68E+04	-.10E+05
16	-.10E+05	-.10E+04
17	-.89E+04	-.10E+05
18	-.10E+05	0.00E+00
19	-.10E+05	0.00E+00
20	-.10E+05	-.10E+05
21	-.10E+05	-.78E+04
22	-.55E+04	-.10E+05
23	-.10E+05	-.78E+04
24	-.10E+05	0.00E+00
25	0.00E+00	0.00E+00
26	0.00E+00	-.10E+05
27	0.00E+00	-.10E+05
28	0.00E+00	-.10E+05
29	0.00E+00	-.10E+05
30	0.00E+00	0.00E+00

THE GRID LOAD FACTOR = 0.80E+04

MAXIMUM MOMENTS AND LOCATION ON LOADED MEMBERS

BETWEEN NODES	MAGNITUDE	LOCATION
HORIZONTAL MEMBERS		
VERTICAL MEMBERS		

** INDICATES > YIELD MOMENT

EXAMPLE 2

GRID CHARACTERISTICS AND LOADING

NUMBER OF NODES HORIZONTAL = 5 HORIZONTAL LENGTH = 10.00
NUMBER OF NODES VERTICAL = 5 VERTICAL LENGTH = 10.00

BOUNDARY CONDITIONS

TOP	SIMPLE SUPORTED
BOTTOM	SIMPLE SUPORTED
LEFT	SIMPLE SUPORTED
RIGHT	SIMPLE SUPORTED

HORZ BEAM PLASTIC BENDING MOMENT = 0.10E+05

VERT BEAM PLASTIC BENDING MOMENT = 0.10E+05

ZX3LP ERROR NUMBER = 0

POINT LOADS ADDED TO THE GRID

REF NODE	DIR FM NODE	DISTANCE FROM NODE	MAGNITUDE	LOAD NUMBER

LINE LOADS

BETWEEN NODES	REF NODE MAG	OTHER NODE MAG	LINE LOAD NUMBER
7 AND 8	1.00	1.00	1
7 AND 12	1.00	1.00	2
8 AND 13	1.00	1.00	3
12 AND 13	1.00	1.00	4

NODE LOADS USED IN CALCULATIONS

NODE NUMBER	NODE LOAD
1	0.00
2	0.00
3	0.00
4	0.00
5	0.00
6	0.00
7	10.00
8	10.00
9	0.00
10	0.00
11	0.00
12	10.00
13	10.00
14	0.00
15	0.00
16	0.00
17	0.00
18	0.00
19	0.00
20	0.00
21	0.00
22	0.00
23	0.00
24	0.00
25	0.00

**** ITERATION NUMBER IS 1 *****

ADDITIONAL CONSTRAINTS USED

AUTOMATIC CONSTRAINTS WERE USED

BETWEEN NODES	LOCATION	TYPE USED

MOMENTS AND LOAD FACTOR

NODE NUMBER	HORIZ MNT	VERT MNT
1	0.00E+00	0.00E+00
2	0.00E+00	0.00E+00
3	0.00E+00	0.00E+00
4	0.00E+00	0.00E+00
5	0.00E+00	0.00E+00
6	0.00E+00	0.00E+00

7	0.89E+04	0.10E+05
8	0.10E+05	0.10E+05
9	0.33E+04	0.67E+04
10	0.00E+00	0.00E+00
11	0.00E+00	0.00E+00
12	0.10E+05	0.10E+05
13	0.10E+05	0.10E+05
14	0.33E+04	0.10E+05
15	0.00E+00	0.00E+00
16	0.00E+00	0.00E+00
17	0.78E+04	0.22E+04
18	0.10E+05	-.11E+04
19	0.98E-03	0.10E+05
20	0.00E+00	0.00E+00
21	0.00E+00	0.00E+00
22	0.00E+00	0.00E+00
23	0.00E+00	0.00E+00
24	0.00E+00	0.00E+00
25	0.00E+00	0.00E+00

THE GRID LOAD FACTOR = 0.18E+03

MAXIMUM MOMENTS AND LOCATION ON LOADED MEMBERS

BETWEEN NODES	MAGNITUDE	LOCATION	
HORIZONTAL MEMBERS			
7 AND 8	0.1170E+05	5.620	**
12 AND 13	0.1222E+05	5.000	**
VERTICAL MEMBERS			
7 AND 12	0.1222E+05	5.000	**
8 AND 13	0.1222E+05	5.000	**

** INDICATES > YIELD MOMENT

**** ITERATION NUMBER IS 2 *****

ADDITIONAL CONSTRAINTS USED

AUTOMATIC CONSTRAINTS WERE USED

BETWEEN NODES	LOCATION	TYPE USED
7 AND 12	5.0	INEQUALITY
8 AND 13	5.0	INEQUALITY
12 AND 13	5.0	INEQUALITY

MOMENTS AND LOAD FACTOR

NODE NUMBER	HORIZ MNT	VERT MNT
1	0.00E+00	0.00E+00
2	0.00E+00	0.00E+00
3	0.00E+00	0.00E+00
4	0.00E+00	0.00E+00
5	0.00E+00	0.00E+00
6	0.00E+00	0.00E+00
7	0.99E+04	0.62E+04
8	0.66E+04	0.89E+04
9	-.17E+04	0.10E+05
10	0.00E+00	0.00E+00
11	0.00E+00	0.00E+00
12	0.61E+04	0.99E+04
13	0.10E+05	0.71E+04
14	0.50E+04	0.10E+05
15	0.00E+00	0.00E+00
16	0.00E+00	0.00E+00
17	0.10E+05	-.68E+02
18	0.10E+05	-.14E+04
19	-.29E-02	0.10E+05
20	0.00E+00	0.00E+00
21	0.00E+00	0.00E+00
22	0.00E+00	0.00E+00
23	0.00E+00	0.00E+00
24	0.00E+00	0.00E+00
25	0.00E+00	0.00E+00

THE GRID LOAD FACTOR = 0.16E+03

MAXIMUM MOMENTS AND LOCATION ON LOADED MEMBERS

BETWEEN NODES	MAGNITUDE	LOCATION
HORIZONTAL MEMBERS		

7 AND 8	0.1058E+05	2.910	**
12 AND 13	0.1000E+05	5.000	
VERTICAL MEMBERS			
7 AND 12	0.1000E+05	5.000	
8 AND 13	0.1000E+05	5.000	

** INDICATES > YIELD MOMENT

EXAMPLE 3

GRID CHARACTERISTICS AND LOADING

NUMBER OF NODES HORIZONTAL = 5   HORIZONTAL LENGTH = 10.00
NUMBER OF NODES VERTICAL = 5   VERTICAL LENGTH = 10.00

BOUNDARY CONDITIONS

TOP	CLAMPED
BOTTOM	CLAMPED
LEFT	CLAMPED
RIGHT	CLAMPED

HORZ BEAM PLASTIC BENDING MOMENT = 0.10E+05

VERT BEAM PLASTIC BENDING MOMENT = 0.10E+05

ZX3LP ERROR NUMBER = 0

POINT LOADS ADDED TO THE GRID

REF NODE	DIR FM NODE	DISTANCE FROM NODE	MAGNITUDE	LOAD NUMBER
7	ON THE NODE	0.00	1.00	1

LINE LOADS

BETWEEN NODES	REF NODE MAG	OTHER NODE MAG	LINE LOAD NUMBER

NODE LOADS USED IN CALCULATIONS

NODE NUMBER	NODE LOAD
1	0.00
2	0.00
3	0.00
4	0.00
5	0.00
6	0.00
7	1.00
8	0.00
9	0.00
10	0.00
11	0.00
12	0.00
13	0.00
14	
16	0.00
17	0.00
18	0.00
19	0.00
20	0.00
21	0.00
22	0.00
23	0.00
24	0.00
25	0.00

**** ITERATION NUMBER IS 1 *****

ADDITIONAL CONSTRAINTS USED

NO ADDITIONAL CONSTRAINTS WERE USED

BETWEEN NODES   LOCATION   TYPE USED

MOMENTS AND LOAD FACTOR

NODE NUMBER	HORIZ MNT	VERT MNT
1	0.00E+00	0.00E+00
2	0.00E+00	-.10E+05
3	0.00E+00	-.10E+05
4	0.00E+00	-.10E+05
5	0.00E+00	0.00E+00
6	-.10E+05	0.00E+00

7	0.10E+05	0.10E+05
8	-.10E+05	0.80E+04
9	-.10E+05	-.10E+05
10	-.10E+05	0.00E+00
11	-.10E+05	0.00E+00
12	0.20E+04	-.10E+05
13	-.10E+05	0.60E+04
14	-.80E+04	-.10E+05
15	-.10E+05	0.00E+00
16	-.10E+05	0.00E+00
17	-.10E+05	-.60E+04
18	-.20E+04	-.10E+05
19	-.10E+05	-.60E+04
20	-.10E+05	0.00E+00
21	0.00E+00	0.00E+00
22	0.00E+00	-.10E+05
23	0.00E+00	-.10E+05
24	0.00E+00	-.10E+05
25	0.00E+00	0.00E+00

THE GRID LOAD FACTOR = 0.80E+04

MAXIMUM MOMENTS AND LOCATION ON LOADED MEMBERS

BETWEEN NODES	MAGNITUDE	LOCATION
HORIZONTAL MEMBERS		
VERTICAL MEMBERS		

** INDICATES > YIELD MOMENT

EXAMPLE 4

GRID CHARACTERISTICS AND LOADING

NUMBER OF NODES HORIZONTAL = 4 HORIZONTAL LENGTH = 10.00
NUMBER OF NODES VERTICAL = 4 VERTICAL LENGTH = 10.00

BOUNDARY CONDITIONS

TOP	SIMPLE SUPORTED
BOTTOM	SIMPLE SUPORTED
LEFT	SIMPLE SUPORTED
RIGHT	SIMPLE SUPORTED

HORZ BEAM PLASTIC BENDING MOMENT = 0.10E+05

VERT BEAM PLASTIC BENDING MOMENT = 0.10E+05

ZX3LP ERROR NUMBER = 0

POINT LOADS ADDED TO THE GRID

REF NODE	DIR FM NODE	DISTANCE FROM NODE	MAGNITUDE	LOAD NUMBER

LINE LOADS

BETWEEN NODES	REF NODE MAG	OTHER NODE MAG	LINE LOAD NUMBER
2 AND 6	1.00	1.00	1
6 AND 10	1.00	1.00	2
5 AND 6	1.00	1.00	3
6 AND 7	1.00	1.00	4

NODE LOADS USED IN CALCULATIONS

NODE NUMBER	NODE LOAD
1	0.00
2	5.00
3	0.00
4	0.00
5	5.00
6	20.00
7	5.00
8	0.00
9	0.00
10	5.00
11	0.00
12	0.00
13	0.00
14	0.00
15	0.00
16	0.00

**** ITERATION NUMBER IS 1 *****

ADDITIONAL CONSTRAINTS USED

AUTOMATIC CONSTRAINTS WERE USED

BETWEEN NODES LOCATION TYPE USED

MOMENTS AND LOAD FACTOR

NODE NUMBER	HORIZ MNT	VERT MNT
1	0.00E+00	0.00E+00
2	0.00E+00	0.00E+00
3	0.00E+00	0.00E+00
4	0.00E+00	0.00E+00
5	0.00E+00	0.00E+00
6	0.10E+05	0.10E+05
7	-.50E+03	0.10E+05
8	0.00E+00	0.00E+00
9	0.00E+00	0.00E+00
10	0.10E+05	0.45E+04

11	0.10E+05	-.98E-03
12	0.00E+00	0.00E+00
13	0.00E+00	0.00E+00
14	0.00E+00	0.00E+00
15	0.00E+00	0.00E+00
16	0.00E+00	0.00E+00

THE GRID LOAD FACTOR = 0.18E+03

MAXIMUM MOMENTS AND LOCATION ON LOADED MEMBERS

BETWEEN NODES	MAGNITUDE	LOCATION	
HORIZONTAL MEMBERS			
5 AND 6	0.1000E+05	10.00	
6 AND 7	9998.	0.1000E-01	
VERTICAL MEMBERS			
2 AND 6	0.1000E+05	10.00	
6 AND 10	0.1034E+05	1.940	**

** INDICATES > YIELD MOMENT

**** ITERATION NUMBER IS 2 *****

ADDITIONAL CONSTRAINTS USED

AUTOMATIC CONSTRAINTS WERE USED

BETWEEN NODES	LOCATION	TYPE USED
5 AND 6	10.	INEQUALITY
6 AND 10	1.9	INEQUALITY

MOMENTS AND LOAD FACTOR

NODE NUMBER	HORIZ MNT	VERT MNT
1	0.00E+00	0.00E+00
2	0.00E+00	0.00E+00

3	0.00E+00	0.00E+00
4	0.00E+00	0.00E+00
5	0.00E+00	0.00E+00
6	0.10E+05	0.10E+05
7	0.13E+04	0.10E+05
8	0.00E+00	0.00E+00
9	0.00E+00	0.00E+00
10	0.10E+05	0.27E+04
11	0.65E+04	0.35E+04
12	0.00E+00	0.00E+00
13	0.00E+00	0.00E+00
14	0.00E+00	0.00E+00
15	0.00E+00	0.00E+00
16	0.00E+00	0.00E+00

THE GRID LOAD FACTOR = 0.18E+03

MAXIMUM MOMENTS AND LOCATION ON LOADED MEMBERS

BETWEEN NODES	MAGNITUDE	LOCATION	
HORIZONTAL MEMBERS			
5 AND 6	0.1000E+05	10.00	
6 AND 7	0.1000E+05	0.1400	**
VERTICAL MEMBERS			
2 AND 6	0.1000E+05	10.00	
6 AND 10	0.1000E+05	1.940	

** INDICATES > YIELD MOMENT

EXAMPLE 5

GRID CHARACTERISTICS AND LOADING

NUMBER OF NODES HORIZONTAL = 3 HORIZONTAL LENGTH = 10.00
NUMBER OF NODES VERTICAL = 3 VERTICAL LENGTH = 10.00

BOUNDARY CONDITIONS

TOP	SIMPLE SUPORTED
BOTTOM	SIMPLE SUPORTED
LEFT	SIMPLE SUPORTED
RIGHT	SIMPLE SUPORTED

HORZ BEAM PLASTIC BENDING MOMENT = 30.

VERT BEAM PLASTIC BENDING MOMENT = 30.

ZX3LP ERROR NUMBER = 0

POINT LOADS ADDED TO THE GRID

REF NODE	DIR FM NODE	DISTANCE FROM NODE	MAGNITUDE	LOAD NUMBER

LINE LOADS

BETWEEN NODES	REF NODE MAG	OTHER NODE MAG	LINE LOAD NUMBER
4 AND 5	1.00	1.00	1

NODE LOADS USED IN CALCULATIONS

NODE NUMBER	NODE LOAD
1	0.00
2	0.00
3	0.00
4	5.00
5	5.00
6	0.00
7	0.00
8	0.00
9	0.00

**** ITERATION NUMBER IS 1 *****

ADDITIONAL CONSTRAINTS USED

AUTOMATIC CONSTRAINTS WERE USED

BETWEEN NODES	LOCATION	TYPE USED

MOMENTS AND LOAD FACTOR

NODE NUMBER	HORIZ MNT	VERT MNT
1	0.00E+00	0.00E+00
2	0.00E+00	0.00E+00
3	0.00E+00	0.00E+00
4	0.00E+00	0.00E+00
5	0.30E+02	0.30E+02
6	0.00E+00	0.00E+00
7	0.00E+00	0.00E+00
8	0.00E+00	0.00E+00
9	0.00E+00	0.00E+00

THE GRID LOAD FACTOR = 2.4

MAXIMUM MOMENTS AND LOCATION ON LOADED MEMBERS

BETWEEN NODES	MAGNITUDE	LOCATION	
HORIZONTAL MEMBERS			
4 AND 5	46.87	6.250	**
VERTICAL MEMBERS			

** INDICATES > YIELD MOMENT

**** ITERATION NUMBER IS 2 *****

ADDITIONAL CONSTRAINTS USED

AUTOMATIC CONSTRAINTS WERE USED

BETWEEN NODES	LOCATION	TYPE USED
4 AND 5	6.3	INEQUALITY

MOMENTS AND LOAD FACTOR

NODE NUMBER	HORIZ MNT	VERT MNT
1	0.00E+00	0.00E+00
2	0.00E+00	0.00E+00
3	0.00E+00	0.00E+00
4	0.00E+00	0.00E+00
5	0.15E+02	0.30E+02
6	0.00E+00	0.00E+00
7	0.00E+00	0.00E+00
8	0.00E+00	0.00E+00
9	0.00E+00	0.00E+00

THE GRID LOAD FACTOR = 1.8

MAXIMUM MOMENTS AND LOCATION ON LOADED MEMBERS

BETWEEN NODES	MAGNITUDE	LOCATION
HORIZONTAL MEMBERS		
4 AND 5	30.00	6.250
VERTICAL MEMBERS		

** INDICATES > YIELD MOMENT

214295

Thesis
H296283 Harvey
c.1
Limit analysis of transversely loaded grillages using linear programming.